

# ABSTRACT

Title of Document: A PROPOSED MECHANICAL-  
METABOLIC MODEL OF THE HUMAN  
RED BLOOD CELL

Stephen Mark Oursler  
Master of Science, 2014

Directed By: Associate Professor Santiago Solares  
Department of Mechanical Engineering

The theoretical modeling and computational simulation of human red blood cells is of interest to researchers for both academic and practical reasons. The red blood cell is one of the simplest in the body, yet its complex behaviors are not fully understood. The ability to perform accurate simulations of the cell will assist efforts to treat disorders of the cell. In this thesis, a computational model of a human red blood cell that combines preexisting mechanical and metabolic models is proposed. The mechanical model is a coarse-grained molecular dynamics model, while the metabolic model considers the set of chemical reactions as a system of first-order ordinary differential equations. The models are coupled via the connectivity of the cytoskeleton with a novel method. A simulation environment is developed in MATLAB® to evaluate the combined model. The combined model and the simulation environment are described in detail and illustrated in this thesis.

A PROPOSED MECHANICAL-METABOLIC MODEL  
OF THE HUMAN RED BLOOD CELL

by

Stephen Mark Oursler

Thesis submitted to the Faculty of the Graduate School of the  
University of Maryland, College Park, in partial fulfillment  
of the requirements for the degree of  
Master of Science  
2014

Advisory Committee:  
Associate Professor Santiago Solares, Chair  
Professor Hugh Bruck  
Associate Professor Teng Li

© Copyright by  
Stephen Mark Oursler  
2014

## Acknowledgements

I would like to express my gratitude to those who have helped me get to the point where I am today. First, I would like to thank my current and former colleagues at the University of Maryland: Hussein Ezzeldin, Alan Wright, Jeffrey Williams, Daniel Ebeling, Adam Kareem, Ben Warner, Babak Eslami, Alfredo Gonzalez, Enrique Lopez, and Sarice Barkley. They have contributed many hours of their own time to the success of this project, from guiding me when I first joined the lab to helping critique my thesis and defense, not to mention taking breaks to share a laugh.

I would like to extend a special thanks to my advisor, Dr. Santiago Solares, at the University of Maryland. He was instrumental in my acceptance into graduate school at the University of Maryland and to completion of my Master's degree. It has been a joy to have Santiago as my advisor and his mentorship over the past three has helped me to mature, both academically and personally, and to produce a thesis that I am proud of.

I would also like to thank the other member of my committee, Dr. Hugh Bruck and Dr. Teng Li. They have both been very understanding and accommodating in the planning of my thesis defense. I would like to thank the U.S. National Science Foundation (NSF) OCI-0904920 for the support they provided me to the project described herein.

I would like to thank my parents. They have made many sacrifices in order to give me educational opportunities and helped foster my interest in science from a young age. I would like to thank my girlfriend, Allison, for

providing inspiration and motivation during this project and especially for being so understanding during the writing of my thesis. I am convinced that I would not have been able to complete this project without her, and I know she is just as happy as I am to see its completion. I promise to not talk about red blood cells for a while. Finally, I would like to thank God. I feel very blessed for the opportunities I have received, and I know that it is only by his grace that I have made it to this point.

# Table of Contents

Acknowledgements.....	ii
Table of Contents .....	iv
List of Figures .....	vii
List of Tables .....	x
Chapter 1: Introduction .....	1
1.1 Motivation.....	1
1.2 Research Objectives.....	3
Chapter 2: Background .....	4
2.1 The Human Red Blood Cell.....	4
2.1.1 The Human Red Blood Cell in Blood Flow.....	4
2.1.2 Structure of the Human Red Blood Cell .....	7
2.1.3 Mechanical Properties of the Human Red Blood Cell.....	11
2.2 Metabolism .....	15
2.3 Continuum Mechanics .....	18
2.4 Molecular Dynamics.....	23
2.5 Statistical Mechanics .....	29
2.5.1 Statistical Mechanics Background.....	29
2.5.2 Polymer Physics.....	31
2.6 Spring Networks and Percolation Theory.....	37
2.7 Triangulation.....	44
Chapter 3: Research Methods and Techniques.....	48
3.1 Triangulation of the hRBC.....	48
3.2 Mechanical Model of the Human Red Blood Cell.....	54
3.2.1 Past Models of the Human Red Blood Cell .....	54
3.2.2 Current Model of the Human Red Blood Cell.....	57
3.2.3 Modifications to CG-hRBC Model.....	65
3.3 Metabolic Model of the Human Red Blood Cell .....	72
3.4 Relationship Between Mechanical and Metabolic Models.....	77
3.4.1 Mechanical-Connectivity Relationship.....	79
3.4.2 Metabolic-Connectivity Relationship .....	82

3.4.3 Mechanical-Metabolic Relationship .....	86
3.4.4 Discrete-Continuum Transformation .....	88
3.4.5 Continuum-Discrete Transformation .....	90
3.4.6 Additional Modification to CG-hRBC Model .....	95
3.5 Simulation of the Human Red Blood Cell .....	98
3.5.1 Simulation Setup .....	100
3.5.2 Energy Minimization .....	100
3.5.3 Simulation Execution .....	101
Chapter 4: Results, Discussion, and Analysis .....	107
4.1 Simulation Environment .....	107
4.2 Triangulation Quality .....	121
4.3 Breaking of Spectrin Links .....	125
4.4 Connectivity as a Function of MgATP Concentration .....	128
4.5 Discussion and Analysis of hRBC Model .....	130
4.5.1 Mechanical Behavior .....	130
4.5.2 Metabolic Behavior .....	134
4.5.3 Mechanical-Metabolic Behavior .....	136
4.5.4 Stress-Free Condition .....	136
4.5.5 Mass Scaling Discrepancy .....	138
4.5.6 Dissipative Forces Error .....	139
4.6 Computational Efficiency Results .....	141
Chapter 5: Conclusions and Future Outlook .....	143
5.1 Intellectual Contributions, Anticipated Benefits, and Progress on Research Objectives .....	143
5.2 Potential Future Work .....	144
5.2.1 Accuracy of Model .....	144
5.2.2 Comprehensiveness of Model .....	151
5.2.3 Computational Improvements .....	154
Appendix .....	157
A.1 Code Files .....	157
A.1.1 f_connectivity.m .....	158
A.1.2 f_cons_forces.m .....	160
A.1.3 f_diss.m .....	167

A.1.4 f_diss_forces.m .....	168
A.1.5 f_ke.m .....	170
A.1.6 f_metabolism.m.....	171
A.1.7 f_plotting.m.....	182
A.1.8 f_pp.m .....	190
A.1.9 f_save.m .....	194
A.1.10 hRBC_Parameter_Derivation.m .....	196
A.1.11 hRBC_Stretching_Simulation.m.....	200
A.1.12 hRBC_Triangulation.m.....	207
A.1.13 vcross.m .....	210
Bibliography .....	211



# List of Figures

<b>FIGURE 1-1:</b> Comparison between heart valves, biological valve [2] on the left and mechanical valve [3] on the right. ....	1
<b>FIGURE 2-1:</b> Classification of some hRBC shapes [19].....	6
<b>FIGURE 2-2:</b> Negative-stain electron microscopy image of membrane skeleton [20].....	7
<b>FIGURE 2-3:</b> Abstraction of the cytoskeleton of the hRBC to a 2D network [21].	8
<b>FIGURE 2-4:</b> Cartoon depiction of the cytoskeleton-lipid bilayer interaction [9]..	9
<b>FIGURE 2-5:</b> Methods of determining cell properties [30]. Micropipette aspiration (c) and optical tweezers stretching (d) are of particular interest to the work performed in this thesis.....	11
<b>FIGURE 2-6:</b> Image from a micropipette aspiration experiment on a hRBC [34].	12
<b>FIGURE 2-7:</b> Depiction of the results from computationally simulating the micropipette aspiration of a hRBC model [36].....	12
<b>FIGURE 2-8:</b> Experimental setup for optical tweezers stretching of the hRBC [39].....	13
<b>FIGURE 2-9:</b> Depiction of the ideal chain model [54].....	32
<b>FIGURE 2-10:</b> Depiction of the WLC model [54].....	36
<b>FIGURE 2-11:</b> Effective medium formulation of diluted spring network used by Feng et al. [74]. ....	42
<b>FIGURE 3-1:</b> Depiction of bending angle between the normal vectors ( $n_1$ and $n_2$ ) of adjacent elements [93].....	59
<b>FIGURE 3-2:</b> Representative area element (RAE) from cytoskeletal network [93] .....	61
<b>FIGURE 3-3:</b> Comparison between WLC-POW-LJ and WLC-POW force-extension relationships.....	68
<b>FIGURE 3-4:</b> Metabolic map of hRBC metabolism [15].....	73
<b>FIGURE 3-5:</b> Mathematical form of hexokinase reaction rate in MMwM model [15].....	74
<b>FIGURE 3-6:</b> Shear modulus as a function of the connectivity of the network. Note that the x-axis is inverted. ....	80
<b>FIGURE 3-7:</b> Connectivity as a Function of MgATP Concentration.....	85
<b>FIGURE 3-8:</b> Shear modulus as a function of MgATP concentration. ....	86
<b>FIGURE 3-9:</b> Comparison between WLC-POW-LJ force-extension relationships for different shear moduli. ....	95

<b>FIGURE 4-1:</b> Visualization of the final configuration of the Combined hRBC model for the example simulation.....	108
<b>FIGURE 4-2:</b> Plot of the contribution to the potential energy from the in-plane term in the force field of the Combined hRBC model as a function of time for the example simulation. ....	109
<b>FIGURE 4-3:</b> Plot of the contribution to the potential energy from the local area term in the force field of the Combined hRBC model as a function of time for the example simulation. ....	110
<b>FIGURE 4-4:</b> Plot of the contribution to the potential energy from the global area term in the force field of the Combined hRBC model as a function of time for the example simulation. ....	111
<b>FIGURE 4-5:</b> Plot of the contribution to the potential energy from the volume term in the force field of the Combined hRBC model as a function of time for the example simulation. ....	112
<b>FIGURE 4-6:</b> Plot the contribution to the potential energy from the bending term in the force field of the Combined hRBC model as a function of time for the example simulation. ....	113
<b>FIGURE 4-7:</b> Plot of the total potential energy of the Combined hRBC model as a function of time for the example simulation.....	114
<b>FIGURE 4-8:</b> Visualization comparing the final configuration of the Combined hRBC model (red) to the initial configuration (black) for the example simulation. ....	115
<b>FIGURE 4-9:</b> Visualization of the final configuration of the Combined hRBC model with dissociated spectrin links (green) for the example simulation.....	116
<b>FIGURE 4-10:</b> Visualization of the vertex strains in the final configuration of the Combined hRBC model for the example simulation. The colorbar on the right side indicates the magnitude of the strains. ....	117
<b>FIGURE 4-11:</b> Visualization of local area strains in the final configuration of the Combined hRBC model for the example simulation. The colorbar on the right side indicates the magnitude of the strains. ....	118
<b>FIGURE 4-12:</b> Visualization of the link strains in the final configuration of the Combined hRBC model for the example simulation. The colorbar on the right side indicates the magnitude of the strains. ....	119
<b>FIGURE 4-13:</b> Visualization of the initial triangulation for the Combined hRBC model, obtained using the isosurface function only.....	122
<b>FIGURE 4-14:</b> Visualization of the final triangulation for the Combined hRBC model, after optimization of the triangulation shown in <b>FIGURE 4-13</b> through use of the <code>distmeshsurface.m</code> function. ....	123

<b>FIGURE 4-15:</b> Visualization of the final configuration of the Combined hRBC model with dissociated spectrin links (green) and broken spectrin links (black) for the breaking links simulation. ....	125
<b>FIGURE 4-16:</b> Visualization of vertex strains in the Combined hRBC model for the breaking links simulation. The colorbar on the right indicated the magnitude of the strains. ....	126
<b>FIGURE 4-17:</b> Plot of the connectivity of the Combined hRBC model as a function of time for the breaking links simulation.....	127
<b>FIGURE 4-18:</b> Plot of connectivity of Combined hRBC model as a function of time for three cases where the MgATP concentration is held fixed at critical values. ....	128
<b>FIGURE 4-19:</b> Plot of connectivity of Combined hRBC model as a function of glucose concentration on a semilog plot. ....	135

## List of Tables

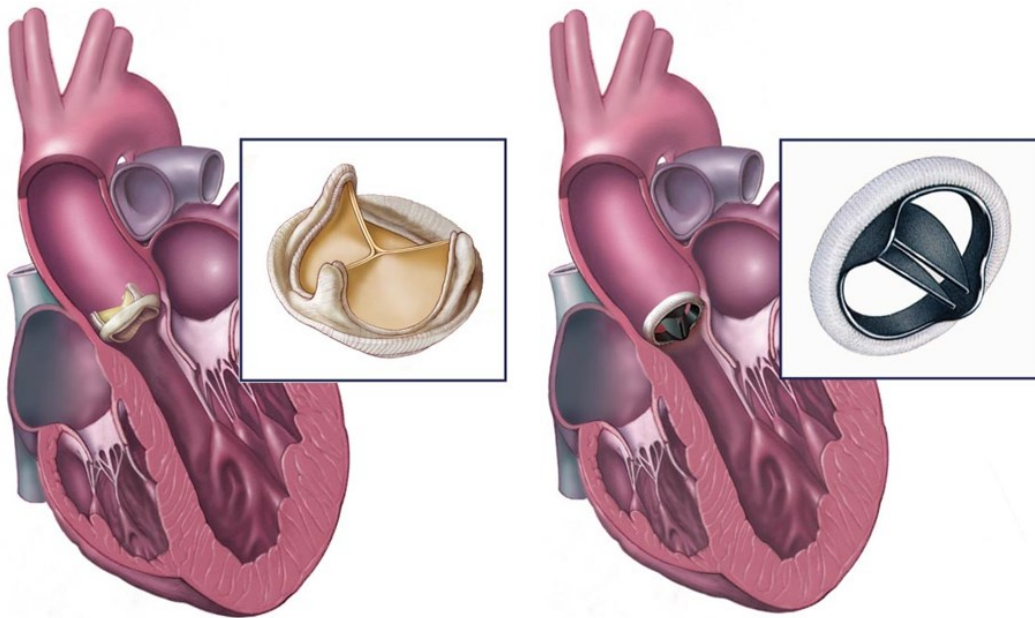
**TABLE 4-1:** Results from triangulation quality analysis. Results for CG-hRBC model taken from [44]. The distribution of link lengths of the Combined hRBC model was determined using **EQUATION 2.13**. The equilateral triangulation quality of the Combined hRBC model was determined using **EQUATION 2.14**. 121

**TABLE 4-2:** Results from mass scaling discrepancy analysis. hRBC density was taken to be  $\rho_{\text{rbc}} = 1.15\rho_{\text{water}}$ . ..... 138

# Chapter 1: Introduction

## 1.1 Motivation

This thesis initially began as an extension of project described in “Multi-scale Modeling of soft matter: Gas Vesicles and Red Blood Cells” by Hussein Ezzeldin [1]. The aim of this project was to incorporate and evaluate metrics for hemolysis of hRBC’s as they travel through artificial (mitral) heart valves, depicted in **FIGURE 1-1**. The motivating hypothesis was that these artificial heart valves change the flow regime from laminar to turbulent, and that this turbulent flow causes hemolysis of the hRBC that results in its premature death.



**FIGURE 1-1:** Comparison between heart valves, biological valve [2] on the left and mechanical valve [3] on the right.

Upon the completion Hussein Ezzeldin’s project, it was decided that a supplemental project was required to further investigate the behavior of the hRBC. This project investigated preexisting models of the hRBC in detail with the intent of combining and modifying these models in order to create a more

complete model of the hRBC that better mimics the behavior observed in blood flow. This complete model, in a similar spirit to other efforts [4, 5, 6], could be used by researchers in multiple fields in order to investigate the behavior of the hRBC under a variety of simulation conditions, such as stretching of the cell.

## 1.2 Research Objectives

The primary objective of this project was to combine current mathematical models of hRBC's in order to create a unified model that more accurately reproduces the behavior of actual hRBC's. Current models of hRBC's either model the mechanical deformations of the cell or the metabolism of the cell, but not both. This thesis is motivated by the hypothesis that the mechanical behavior and the metabolism of the hRBC are coupled, and cannot be modeled separately. The combined model uses coupling conditions to link the mechanical behavior to the metabolism to define their interaction and simulate them together. Ideally, when they are simulated together, the results obtained are more accurate than the individual simulations alone.

The secondary objective of this project, after the creation of a unified model, is to implement the model in MATLAB® [7] so that simulations on the model can be performed. The implementation of the coded model would then become a simulation environment. This environment can be used by researchers to gain a better understanding of hRBC behavior in a variety of conditions. Ideally, simulations could be used to confirm of experimental results, and to predict hRBC behavior in conditions that have not been experimentally tested yet.

The tertiary objective of this project is to increase interest in modeling and simulation of human red blood cells in the community. The hope is that the creation of a unified model of the hRBC and the implementation of the model into a simulation environment that is easily accessible to researchers will renew interest in the investigation of the hRBC.

## Chapter 2: Background

Chapter 2 is intended to refresh the reader's memory regarding several concepts that form the basis of mathematical modeling and computational simulation used in this thesis. It is also intended to introduce sufficient background information regarding the human red blood cell (hRBC) itself, as well as information specific to the modeling/simulation of the hRBC, to readers not familiar with the topic. Chapter 2 is not a comprehensive review of the extremely extensive history of hRBC modeling/simulation, and number of important milestones in the history of hRBC modeling/simulation are not mentioned. A more comprehensive review (up to 2006) is provided by Lim et al. [8].

### 2.1 The Human Red Blood Cell

#### 2.1.1 The Human Red Blood Cell in Blood Flow

The hRBC is one of the main components of blood and it is critical to the transport of oxygen and carbon dioxide in the body. It makes up about 45% of blood by volume. The measurement of the concentration of red blood cells is known as hematocrit. The other major solid constituents of blood are white blood cells and platelets. The fluid phase of blood is called plasma. The hRBC (also known as an erythrocyte) is anucleate, meaning it does not have a nucleus. In addition, it lacks the majority of the organelles found in most other eukaryotic cells. As a consequence, the hRBC is unable to perform many of the normal regulatory functions of eukaryotic cells. This results in a finite lifespan of around 120 days in circulation, significantly less than other cells. Upon the completion of its life in



circulation, the hRBC is recycled in the spleen [9, 10]. It is important to note, however, that the hRBC does possess an active metabolism in its cytoplasm that assists in some functions of cell upkeep, such as glycolysis and ion transport across the membrane [11, 12, 13, 14, 15].

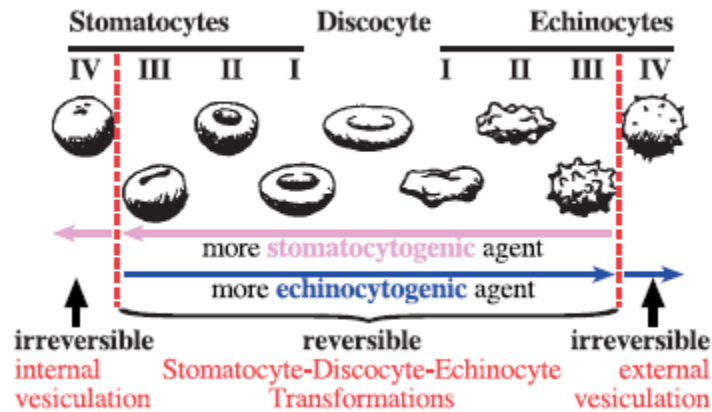
The lack of organelles is necessary to the hRBC being able to fulfil its role in the body, however, because it makes space for the protein hemoglobin. Hemoglobin makes up about 70% of the dry weight of a hRBC, and it is the physical carrier of the oxygen molecule. In order to fulfill this role, the hRBC has to be highly deformable. Under normal conditions, a hRBC is a biconcave discoid with a diameter of about 8  $\mu\text{m}$ , a height of about 2.5  $\mu\text{m}$ , a surface area of about 135  $\mu\text{m}^2$ , and a volume of about 94  $\mu\text{m}^3$  [9, 16]. To effectively deliver oxygen to tissues, the hRBC has to significantly deform into an ellipsoid-like shape in order to transit through capillaries as narrow as 2  $\mu\text{m}$ . The hRBC is able to assume its original shape upon exiting the capillary [17]. Part of their significant deformability is due to the surface area-to-volume ratio (a sphere with equal surface area would have a volume of about 147  $\mu\text{m}^3$ ). The excess surface area is also beneficial to hRBC function, providing a significantly larger surface for oxygen diffusion than a sphere of the same volume would provide. The deformability of a hRBC is governed by three main factors [11, 18]:

- 1) The geometry of the cell (specifically, the surface area-to-volume ratio)
- 2) The rheological properties of the hRBC cytoplasm (specifically viscosity)
- 3) The rheological properties of the cell membrane (specifically viscosity)

In addition to providing the hRBC itself with some interesting mechanical and rheological responses, the hRBC has a significant impact on blood flow.

Blood flow displays non-Newtonian behavior, which means the apparent viscosity of blood varies with applied shear stress. This non-Newtonian behavior is primarily due to the presence of hRBC's. Blood is considered to be a two phase suspension, with the hRBC's viewed as either the solid suspended phase or an additional liquid phase (since the majority of the volume of a healthy hRBC is the cytoplasm). Since hRBC's make up such a large percentage of blood volume, their effect on blood flow is significant. The study of blood flow is known as hemorrheology or hemodynamics [18].

The hRBC can display a wide variety of shapes as a function of the conditions the cell is subject to, as depicted in **FIGURE 2-1**.



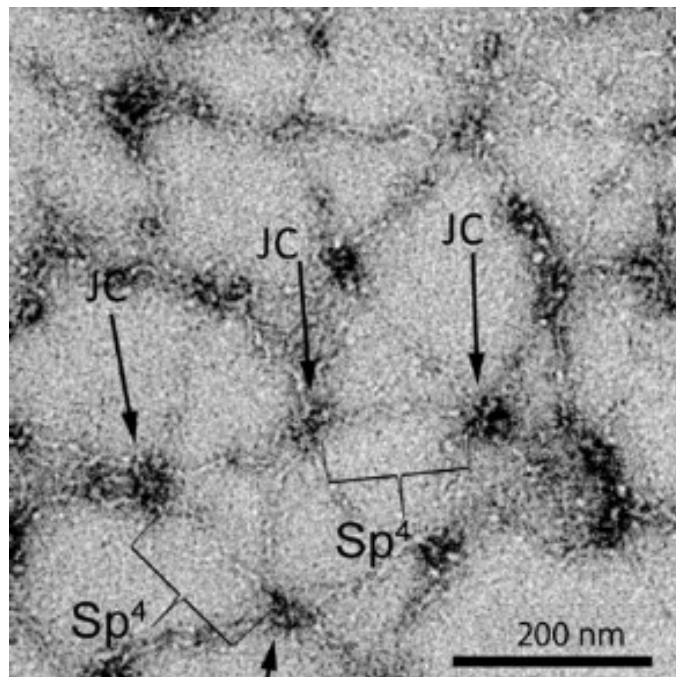
**FIGURE 2-1:** Classification of some hRBC shapes [19].

As long as the deformations of the hRBC are sufficiently small, the Stomatocyte-Discocyte-Echinocyte (SDE) transformation is reversible. Significant enough deformations of the hRBC, however, lead to vesiculation of the cell and can cause permanent damage. The shapes in **FIGURE 2-1** are just a subset of the shapes observed in experiments (not all shapes are stable, however) [19].

## 2.1.2 Structure of the Human Red Blood Cell

Compared to other cells found in the human body, the hRBC appears relatively simple. Lacking organelles (as stated previously in [SUBSECTION 2.1.1](#)), the primary components of the hRBC are hemoglobin, cytoplasm, the cytoskeleton, and the cell membrane. Of particular interest to the mechanical modeling and simulation of the hRBC are the cytoskeleton and the cell membrane.

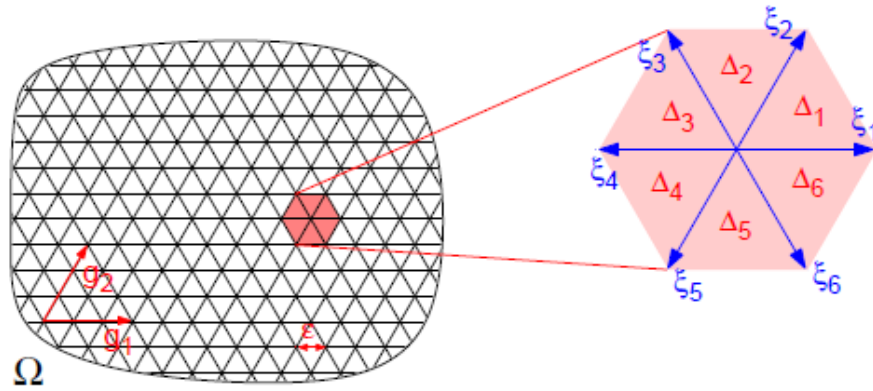
The cytoskeleton serves to provide support and structure to the hRBC to ensure proper morphology. The cytoskeleton is located on the inner face of the cell membrane (inside the hRBC, in the cytoplasm). It forms a disordered, two-dimensional (2D), triangular network, as can be seen in [FIGURE 2-2](#).



***FIGURE 2-2:** Negative-stain electron microscopy image of membrane skeleton [20]*

The links are the physical connection between the nodes in the network. The nodes in the network are the locations where the links meet and the cytoskeleton

is anchored to the cell membrane. The links in the cytoskeletal network (indicated by “SP<sup>4</sup>” in **FIGURE 2-2**) are a protein known as spectrin. The nodes in the cytoskeletal network (indicated by “JC” in **FIGURE 2-2**) are another protein known as actin. A representation of this network is shown in **FIGURE 2-3**.

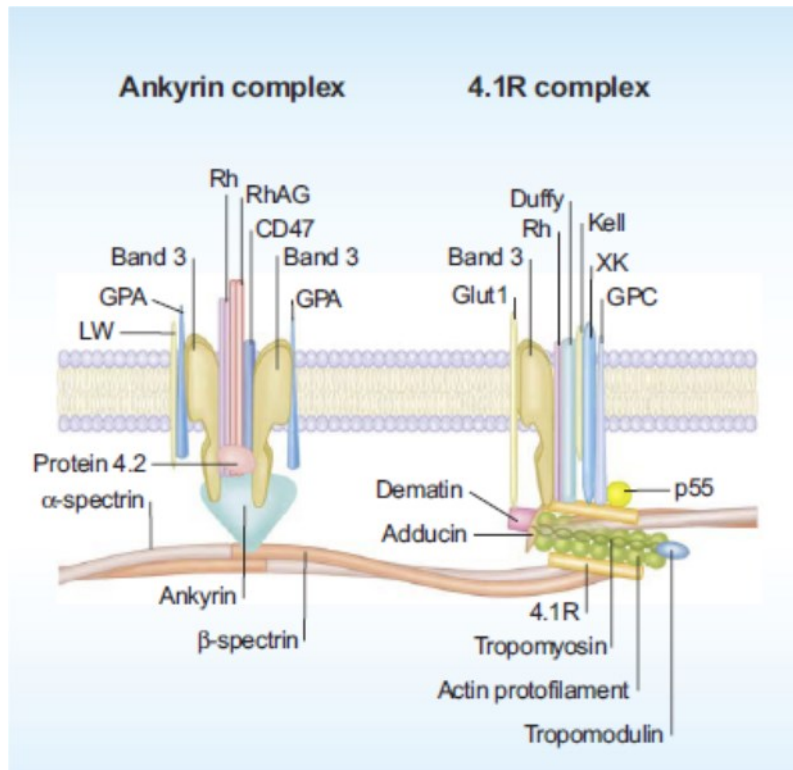


**FIGURE 2-3:** Abstraction of the cytoskeleton of the hRBC to a 2D network [21].

In a perfect triangular network, each node in the network has exactly six links connected to it. The degree of a network is the average number of links per node in the network. The degree of a single node in the network, however, is simply the number of links connected to that particular node. A perfectly triangular network is of degree-6, since each node has exactly six links connected to it. Not all degree-6 networks are triangular networks, however, since the degree is an average quantity. A network with an equal number of nodes with five links and nodes with seven links would still be a degree-6 network, but it is clear that such a network is not perfectly triangular. The cytoskeletal network of the hRBC is not a perfect network. Imaging of the hRBC cytoskeleton suggests that the actual degree of the cytoskeletal network is somewhere between 4.2 and 5.5 [22].

Spectrin is a long, filamentous protein that actually consists of the head-to-head association of two antiparallel heterodimers,  $\alpha$ -spectrin and  $\beta$ -spectrin.  $\alpha$ -spectrin has 22 triple-helical repeats of 106 amino acids and a molecular weight of about 280 kDa, while  $\beta$ -spectrin had 17 triple-helical repeats of 106 amino acids and a molecular weight of about 246 kDa. Actin is another filamentous protein. It has a molecular weight of about 40 kDa. It is sometimes referred to as a protofilament because it is shorter than spectrin [9, 10, 23, 22, 24, 25].

The actual structure of the cytoskeleton is considerably more complicated than what was described above. A more complete picture of the interaction is can be seen in [FIGURE 2-4](#). Additional detail on the structure is provided by Burton et al. [26].

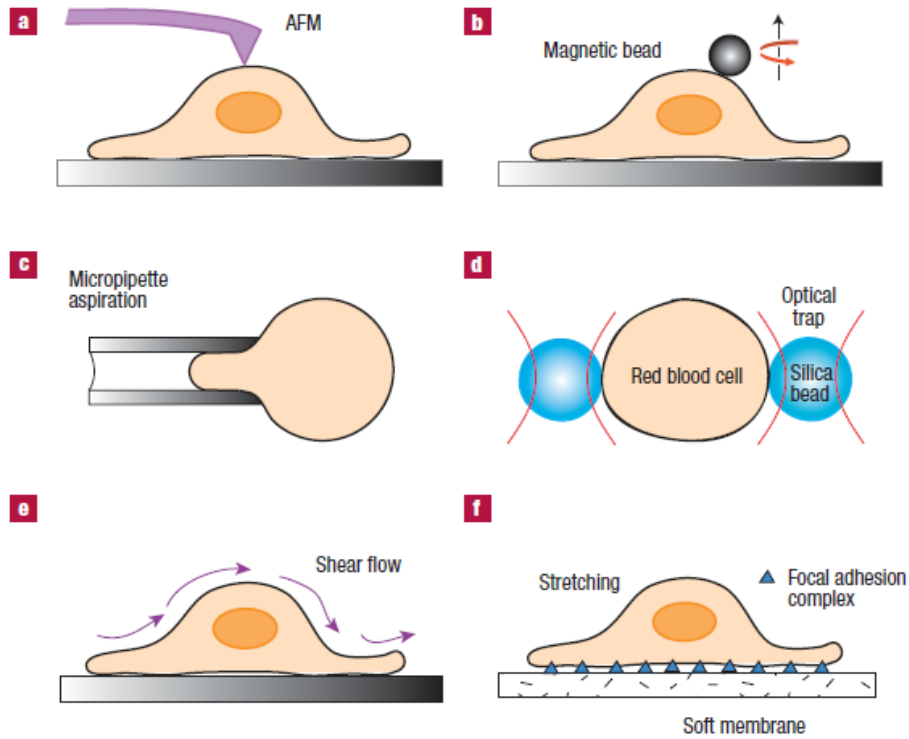


*FIGURE 2-4: Cartoon depiction of the cytoskeleton-lipid bilayer interaction [9]*

The cell membrane serves as a boundary between the cytoplasm and the blood plasma. It consists two sheets made up primarily by lipids. For this reason, it is often referred to as a lipid bilayer. The lipids self-assemble so that the hydrophobic tails of each sheet are facing those of the other sheet, leaving only the heads of the lipid molecules exposed to the cytoplasm/blood plasma. There are a number of integral membrane proteins located in the lipid bilayer that provide the attachment points between the bilayer and the cytoskeleton. Some of these integral proteins also serve as active transport channels. These channels consume chemical energy in order to actively transport larger molecules in or out of the hRBC. Smaller molecules, such as oxygen, can passively diffuse through the lipid bilayer. An understanding of the hRBC structure is important because it directly impacts the properties and behavior of the hRBC [27, 28].

### 2.1.3 Mechanical Properties of the Human Red Blood Cell

The mechanical properties of the hRBC can be measured by a variety of different means [29]. Several of the common measurement techniques are shown in **FIGURE 2-5**.

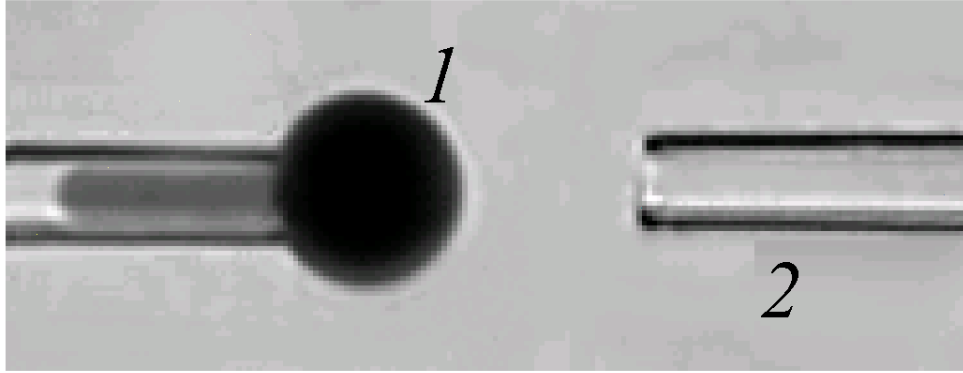


**FIGURE 2-5:** Methods of determining cell properties [30]. Micropipette aspiration (c) and optical tweezers stretching (d) are of particular interest to the work performed in this thesis.

A wide variety of mechanical properties can be measured using these techniques.

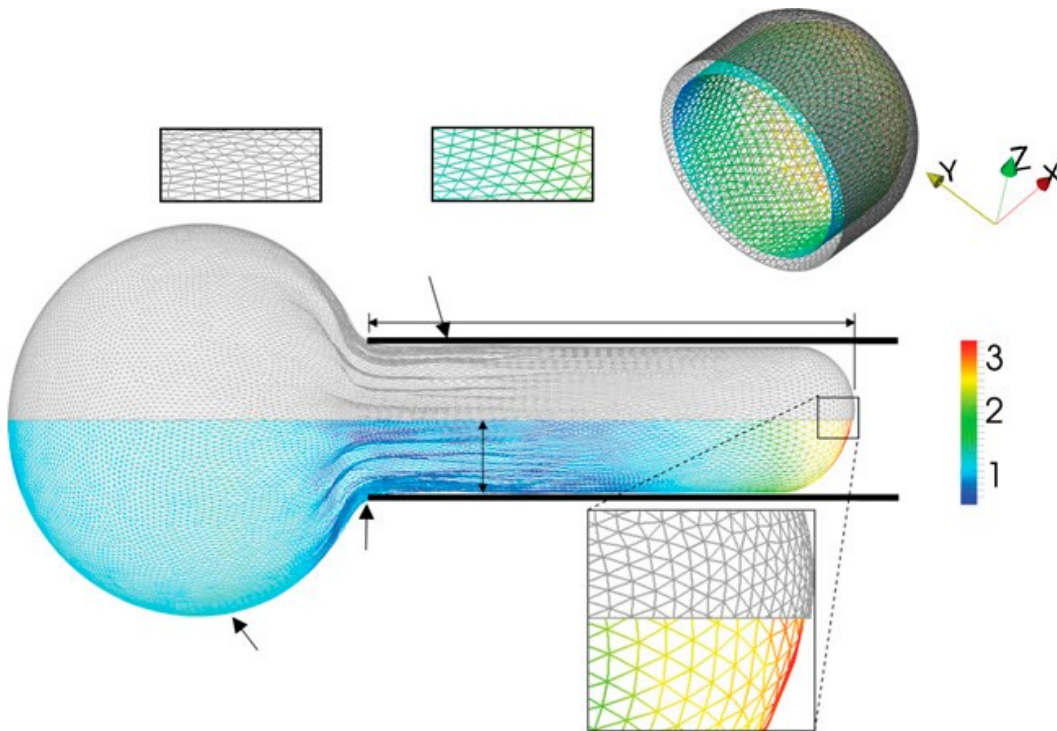
A number of these measurement techniques are described in more detail by Leckband et al. [31]. Two of these techniques are of particular interest to the work performed in this thesis, and will be discussed in more detail below.

Micropipette aspiration used to be the primary method of determining hRBC cell properties [32, 33]. In this measurement technique, part of the hRBC is aspirated into a micropipette, as illustrated in **FIGURE 2-6**.



*FIGURE 2-6: Image from a micropipette aspiration experiment on a hRBC [34].*

The length of the aspirated portion is recorded using optical microscopy. Using the diameter of the micropipette tube and the aspiration pressure, elastic properties can be extracted using constitutive relationships [35]. A depiction of the results from computationally simulating the micropipette aspiration of a hRBC model is illustrated in **FIGURE 2-7**.

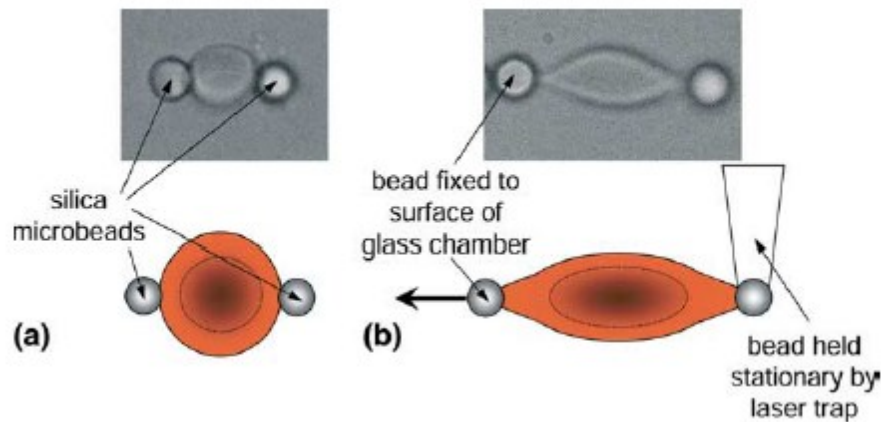


*FIGURE 2-7: Depiction of the results from computationally simulating the micropipette aspiration of a hRBC model [36].*



The micropipette aspiration measurement technique has lost some popularity in recent years because the results from aspiration experiments can be rather inaccurate, depending upon experimental conditions. The aspirated portion of the hRBC is subject to very uneven deformations and severe bending in both the cap and entrance regions. Together, these effects can produce significant errors in the experimental results [37, 38]. An alternative measurement technique, known as optical tweezers stretching, can deform the hRBC more evenly during experiments, potentially providing more accurate mechanical property data.

In the optical tweezers measurement technique, silica microbeads are attached to the hRBC at diametrically opposed points. One bead is anchored to part of the experimental setup that will apply stretching forces. The other bead is “trapped” using a laser beam. The experimental setup is illustrated in [FIGURE 2-8](#).



*FIGURE 2-8: Experimental setup for optical tweezers stretching of the hRBC [39].*

When a stretching force is applied to the anchored node (the left node in [FIGURE 2-8](#)) the intensity of the laser beam is increased so that trapped node (the right node in [FIGURE 2-8](#)) remains stationary. Another benefit to the optical tweezers

measurement technique is that very large stretching forces can be applied using this stretching method. Mechanical property data from optical tweezers stretching experiments on the hRBC can be found in [40].

The optical tweezers measurement technique [32, 33, 39, 41, 42, 43] is of particular interest in this thesis. This measurement technique can be “mimicked” in simulations of the hRBC [44] in order to determine the corresponding mechanical properties of the hRBC model. At a minimum, these computational measurements performed on a hRBC model must produce the same results as the measurements performed on the actual hRBC for the hRBC model to be valid. This mechanical property data is obtained by evaluating the response of the hRBC undergoing these measurements. For the hRBC model to be accurate, it should have the same response as the actual hRBC when subject to the same measurement. In general, an understanding of both the structure of the hRBC and its properties is of the utmost importance in the efforts of modeling the cell. The Structure-Property-Function relationship is ubiquitous in biology in general, and provides guidance in the modeling of biological systems [45].

## 2.2 Metabolism

A metabolism is the set of chemical reactions that support the life of an organism. The participants of a chemical reaction are commonly known as metabolites. In a metabolism, the products of the previous reaction become the reactants of the next reaction. Reactions can be either reversible or irreversible (depending upon the specific metabolites involved) and either spontaneous or nonspontaneous (depending upon the free energy change of the reaction). Rates of individual reactions can depend on the concentration of metabolites who do not directly participate in the reaction (such as enzyme reactions).

The dependence of reaction rates on metabolite concentrations allows for regulation of the metabolism (either as a whole or just a specific part, depending of the specific structure of the metabolism). Since reaction rates tend to increase with increasing reactant concentrations (and decrease with increasing product concentrations), the metabolism exhibits negative feedback that allows for self-regulation. A metabolism that is not subject to any external influences will reach a homeostatic state (steady-state condition).

The rate of a chemical reaction can be mathematically modeled using a first-order, ordinary differential equation (ODE). The concentration of a metabolite participating in the reaction as a function of time can be obtained from the solution to the reaction rate ODE. Combining the mathematical representations of reactions rates into a system of equations allows for a metabolism to be modeled as a coupled set of first-order ODE's. Because an individual reaction can depend on the concentrations of other metabolites (which

are varying), the ODE representing the reaction rate is often nonlinear. Because of this nonlinearity, an analytical solution of the coupled set of ODE's is generally not possible and numerical techniques must be used.

The concentrations of the metabolites cannot be negative at any point during the simulation because a negative concentration is physically impossible. If the concentration of a metabolite decreases to less than zero during a simulation, the simulation itself will often become unstable. The non-negative metabolite concentration requirement is extremely important in the mathematical modeling of chemical reactions, but it is rarely mentioned because it is considered trivial. The non-negative requirement is easily enforced when the metabolism contains a small number of reactions that can be modeled analytically. When the number of reactions and/or the complexity of the rate equations increases to the point where numerical methods are needed, enforcing the non-negative requirement on the concentrations becomes more difficult. Simply adding provisions to the metabolic simulation that set negative metabolite concentration back to zero violate the mass balance of the system. The parameters in the metabolism itself must be adjusted to ensure that negative metabolite concentrations do not occur. A method of checking the concentrations during simulation should also be implemented to halt unstable simulations [46].

The phenomenon of stiffness is often observed when numerical methods are used to solve the ODE's used to model chemical reactions. The definition of stiffness is qualitative and based on observed behavior of the numerical solution rather than precise mathematical terms. A stiff equation is a differential equation

that exhibits numerical instability when certain numerical methods applied to solve the equation, unless an extremely small time step is used. This instability is present even when the solution is not varying rapidly, and therefore is a property of the differential system itself. The stiffness observed in the ODE's used to model chemical reactions is often attributed to the widely varying values of the parameters and variables in the equations (sometimes orders of magnitude different from one another). As a result, numerical methods that are designed to handle stiff equations must be used. These numerical methods often combine explicit and implicit numerical integration schemes to obtain a solution [15].

## 2.3 Continuum Mechanics

Continuum mechanics (CM) forms the basis of the majority of the methods used to describe the kinematics and mechanical behavior of the hRBC. The main quantities of interest in CM are stress, strain, and displacement. Both stress and strain are field quantities (second order tensors), while displacement is a vector quantity. The majority of the information presented in this section is borrowed from a solid mechanics text [47] and a cell mechanics text [48]. As an aside, the cell mechanics text is an excellent didactic tool for illustrating the application of CM principles to biological materials, specifically the cytoskeleton and cell membrane.

The fundamental assumption of CM is that the object of interest is a continuous distribution of matter and that the matter that makes up the object completely fills the region of space that the object occupies. This allows the object to be modeled as a continuum and, after the application of fundamental physical laws (e.g. conservation of mass, conservation of linear momentum, conservation of angular momentum, and conservation of energy) and constitutive equations (e.g. Hooke's law), differential equations describing the behavior of the object can be derived and engineering analyses can be performed on it. The continuum assumption appears incorrect upon first glance because it neglects the fact that all objects are made up of atoms. However, at length scales considerably larger than the atomic scale, this assumption becomes more valid.

The continuum assumption hinges upon the concept of a representative volume element (RVE). A RVE is the smallest volume of the material that has

properties/behavior identical to that of the bulk material (measurements performed on the RVE yield results identical to the same measurements performed upon the bulk material). At scales larger than the RVE, the inherent heterogeneity of the microstructure of the material can be ignored and the material can be treated as an equivalent homogeneous material. At scales smaller than the RVE, the continuum assumption is no longer valid. If a continuum description of the material is needed at scales smaller than the RVE, a statistical volume element (SVE) needs to be defined. The properties of a SVE are defined using probability theory, leading to random continuum fields. The concept of the SVE will be discussed later in [SECTION 2.5](#).

Another assumption commonly made in CM analysis (and hRBC analysis) is that the object in question is a Cauchy elastic material. Assuming an object is a Cauchy elastic material has several important implications in the stress analysis of the object. The first implication is that the stress is independent of the deformation path and the deformation history (stress is a state function) as well as the rate of deformation (no viscous/viscoelastic behavior). It is important to note that, even though the stress is independent of the deformation path, the work done by the stress may not be independent of the deformation path (a Cauchy Elastic material is not necessarily conservative). The second implication is that the stress and constitutive relations are spatially local, which means that the stress at a point is a function of only the deformation at that point (not the total deformation or motion of the object). The third implication is that the properties of the object are independent of body forces and inertial forces. The fourth implication is that the

object is homogeneous, which means that the properties do not vary as a function of location in the object.

A number of these assumptions are made in CM analyses of the hRBC, either explicitly stated or implicitly assumed through the use of certain equations. The use of these assumptions in a hRBC model can have a significant impact of the results of a simulation. An understanding of the implications resulting from these assumptions is very important in ensuring that a CM analysis of the hRBC is physically meaningful.

Relationships between the main quantities of interest in CM can be derived. Strain is related to displacement (the strain-displacement relation) by:

$$\varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial u_j} + \frac{\partial u_j}{\partial u_i} - \frac{\partial u_k}{\partial u_i} \frac{\partial u_k}{\partial u_j} \right) \approx \frac{1}{2} \left( \frac{\partial u_i}{\partial u_j} + \frac{\partial u_j}{\partial u_i} \right) \quad (2.1)$$

where  $u$  is the displacement. The second part of [EQUATION 2.1](#) is an approximation that is commonly used when deformations are small, since it results in simpler calculations. The stress field is independent of the material in question (but strain and displacement are not). Stress can be related to strain through a constitutive relationship.

Under the assumption of linear elasticity (which assumes that stress is linearly proportional to strain), the constitutive relation between stress and strain is given by:

$$\sigma_{ij} = C_{ijkl} \varepsilon_{kl} \quad (2.2)$$

where  $\sigma$  is the stress,  $C$  is the stiffness tensor, and  $\varepsilon$  is the strain. This linear relationship is only valid for small deformations. Since stress and strain are second order tensors, the stiffness tensor is a fourth order tensor. Due to



symmetry requirements in the stress and strain (due to conservation of angular momentum) and existence requirements on the strain energy density, there are only 21 independent elastic constants in the stiffness tensor (out of 81 total elastic constants) for a fully anisotropic material (material properties vary with direction). If symmetries are present in the material itself, the number of independent elastic constants can be reduced further. The simplest case is that of an isotropic material (material properties do not vary with direction). An isotropic material only has two independent elastic constants. Despite the fact that no real-world materials are truly isotropic, the isotropy assumption is commonly made in CM analysis (e.g. in the continuum modeling of the hRBC) because it greatly simplifies the stress analysis.

A nonlinear elastic formulation for the constitutive relation between stress and strain can be used for materials subject to deformations large enough that the linear elastic assumption is no longer valid. Under the assumption that constitutive relation between stress and strain can be derived from a strain energy density function. These materials are known as hyperelastic, and the stress-strain relationship is given by:

$$\mathbf{S} = \frac{\partial W}{\partial \mathbf{E}} \quad (2.3)$$

where  $W$  is the strain-energy density function. Rubber is often modeled as a hyperelastic material. A hyperelastic material is a special case of a Cauchy elastic material. Note that a hyperelastic material is conservative, meaning that all of the strain energy can be recovered when the applied loads are removed from the object and it is allowed to return to its reference configuration. A linear elastic

material is a special case of a hyperelastic material, and therefore is conservative as well. The concept of a hyperelastic material will be revisited later in

### SUBSECTION 3.2.1.

Under the assumptions of linear elasticity and isotropy, relationships between the elastic constants can be derived:

$$Y = \frac{9K\mu}{3K+\mu} \quad (2.4)$$

$$\nu = \frac{3K-2\mu}{2(3K+\mu)} \quad (2.5)$$

$Y$  is Young's modulus,  $\mu$  is the shear modulus,  $K$  is the bulk modulus, and  $\nu$  is Poisson's ratio. The shear modulus and the bulk modulus must be positive. This requirement results in bounds on Poisson's ratio:

$$-1 \leq \nu \leq \frac{1}{2} \quad (2.6)$$

Sometimes, for simplification reasons, an object is assumed to be incompressible. For an incompressible, three-dimensional (3D) object:

$$\nu = \frac{1}{2}$$

$$K = \infty$$

$$Y = 3\mu \quad (2.7)$$

The strains in an incompressible object are not completely independent from one another, since the volume of the object must be preserved under deformation. In stress analysis, the incompressibility assumption is used to reduce the number of degrees of freedom of the object, streamlining the analysis.

## 2.4 Molecular Dynamics

Molecular dynamics (MD) is a computational method of simulating the motion of interacting particles in an N-body problem. MD is used in this thesis to simulate the hRBC. MD envisions the particles of interest in the system as point masses and allows them to interact. The interaction between particles is governed by a predefined set of rules. The interaction between particles results in forces acting on the particles, which in turn result in accelerations, and ultimately motions of the particles. The motions of the particles are governed by classical mechanics (Newton's laws of motion). The force acting on each particle is the negative gradient of the interaction energy with respect to separation distance. The force due to the interaction takes the form:

$$\vec{F}_i = -\frac{\partial U(r_{ij})}{\partial \vec{r}_{ij}} = -\vec{\nabla}U(r_{ij}) \quad (2.8)$$

where  $U(r_{ij})$  is the interaction energy between particle  $i$  and particle  $j$ ,  $\vec{r}_{ij}$  is the vector between the particles,  $r_{ij}$  is the magnitude of  $\vec{r}_{ij}$ ,  $\vec{F}_i$  is the force acting on particle  $i$  as a result of the interaction, and  $\vec{\nabla}$  is the gradient operator. The force on particle  $j$  will have the same magnitude as the force on particle  $i$ , but will act in the opposite direction ( $\vec{F}_j = -\vec{F}_i$ ) due to Newton's third law. The acceleration of a particle due to this force can be found using Newton's second law:

$$\vec{F}_i = m_i \vec{a}_i = m_i \frac{d^2 \vec{r}_i}{dt^2} = m_i \ddot{\vec{r}}_i \quad (2.9)$$

where  $m_i$  is the mass of particle  $i$  and  $\vec{a}_i$  is the acceleration of particle  $i$ , which is the second derivative of the position of the particle,  $\vec{r}_i$ , with respect to time.

In the ideal case, the interaction energy goes to zero as the separation distance goes to infinity (no interaction at large separations) and the interaction energy goes to infinity as the separation distance goes to zero (steric repulsion to prevent particle overlap). There exists an equilibrium separation distance between two particles where the attractive and repulsive portions of the interaction energy between the particles balance each other out and neither particle experiences a force as a result of the interaction between the two. At this equilibrium separation distance, the interaction energy is at a minimum.

For most systems, Newton's laws of motion need to be integrated numerically. Often, the Verlet algorithm is used to calculate the trajectories of the particles. The Verlet algorithm is a type of finite difference method that is used to numerically integrate second order ordinary differential equations. It offers greater stability than the simple Euler method with no additional computational cost. The basic form of the Verlet algorithm is:

$$\vec{r}_{t+\Delta t} = \vec{r}_t + \vec{v}_t \Delta t + \frac{1}{2} \vec{a}_t \Delta t^2 \quad (2.10)$$

where  $t$  is the current time step,  $\Delta t$  is the duration of the time step,  $\vec{r}_t$  is the position of the particle at the beginning of the time step,  $\vec{v}_t$  is the velocity of the particle at the beginning of the time step,  $\vec{a}_t$  is the acceleration of the particle at the beginning of the time step, and  $\vec{r}_{t+\Delta t}$  is the position of the particle at the end of the time step. An initial position and velocity ( $\vec{r}_0$  and  $\vec{v}_0$ ) must be provided at the start of the integration. [EQUATION 2.10](#) is for a single particle (the particle identifier subscript was left off to prevent the equation from becoming congested).

An alternate version of the Verlet algorithm exists (known as Störmer's method).

The basic form of Störmer's method is:

$$\vec{r}_{t+\Delta t} = 2\vec{r}_t - \vec{r}_{t-\Delta t} + \vec{a}_t\Delta t^2 \quad (2.11)$$

**EQUATION 2.11** is also for a single particle. Note that the velocity of the particle does not appear in Störmer's method. This is because the velocity term introduces error into the calculation. The local error in the position is  $O(\Delta t^4)$ , but the local error in velocity is  $O(\Delta t^2)$ , using big-O notation. This means that Störmer's method is two orders of magnitude more accurate than the basic Verlet algorithm. Additional calculations, however, must be performed if the velocity is to be obtained from Störmer's method, and the local error in the velocity obtained this way will be  $O(\Delta t^2)$  [49]. The numerical integration method used in this thesis is explained later in **SECTION 3.5**.

The concept of a force field is critical to MD analysis and simulation. A force field is the set of potential energy functions that define the interaction between the particles (the potential energy contribution to the Hamiltonian of the system). These interactions generally included two-body, three-body, and four-body interaction potentials as well as non-bonded interaction potentials. The exact form of and values of the parameters in the potential energy functions is usually determined experimentally. In some cases, high-level quantum mechanical calculations can be used to determine the form and parameter values. One of the major assumptions made about the interaction potentials is that they are spherically symmetric, which means that the magnitude of the interaction potential is only a function of the separation distance between the particles (it

does not vary with the orientation of the particles with respect to one another). This assumption is only accurate for the most simple of systems (e.g. charged particles in a vacuum). In most systems, especially systems with chemical bonds, the interaction potentials are directionally dependent.

A direct relationship between MD and a 2D spring network can be obtained for a 2D system with only two-body interaction potentials between the particles. In this case, the springs in the network directly correspond to the two-body interaction potentials. The two-body interaction potentials are sometimes referred to as in-plane interactions. This relationship is important in the context of hRBC modeling because, during the formulation of the hRBC model, the membrane of the hRBC is visualized as an infinite, 2D spring network. The concept of a spring network is revisited in [SECTION 2.6](#).

MD is often used for simulation on an atomic scale. The particles in these simulations are typically the atoms that make up the system of interest. The complexity of a MD system increases nonlinearly with the number of particles in the system. This can result in a simulation that is very computationally intensive, even for a small increase in the number of atoms in the system. There are a number of “workarounds” that help alleviate the computational expense. Of most significance to this thesis is the method of coarse-graining.

Coarse-graining is a method of reducing the number of degrees of freedom of a system by treating a subset of the particles as a single particle. This reduces the effective number of particles in the simulations, which in turn reduces the number of interactions between particles that need to be calculated. Depending

on the level of coarse-graining, simulation times can be orders of magnitude shorter. The tradeoff of coarse-graining is that some of the finer details of the dynamics of the system are lost as a result of the reduction in the number of degrees of freedom. The model of the hRBC presented in [SUBSECTION 3.2.2](#) is highly coarse-grained as compared to a system modeled on the atomic scale. The loss of resolution is acceptable in the case of the hRBC model because the motions of the individual atoms relative to each other do not affect the overall behavior of the hRBC to a significant degree.

It is important to note that many systems required a potential energy minimization procedure before MD simulations of system can be performed. This procedure consists of performing a steepest descent energy minimization simulation of the system under no external interactions at zero Kelvin. The steepest descent (also known as gradient descent) simulation seeks the global potential energy minimum, which is a function of the configuration of the system. Performing the minimization at zero Kelvin ensures that the system will remain in its minimum potential energy configuration. The zero Kelvin condition is enforced by setting the velocity of all particles in the system to zero (since temperature is a measure of the average kinetic energy of the system) after each time step. The drawback of the steepest descent minimization is that the potential energy asymptotically approaches its global minimum value, but never actually reaches it. Because of this, an energy threshold has to be defined and the minimization is run until the changes in the potential energy between time steps are less than this threshold value [49].

There is a variant of MD known as steered molecular dynamics (SMD). In SMD, the system is probed by an external force to examine its response. The external force is applied in such a way that mimics probing of the actual system (e.g. stretching of a folded protein to examine its unfolding behavior). SMD is important in the context of hRBC simulations because the mechanical properties of the hRBC are obtained through probing of the hRBC and examination of its response. Likewise, the mechanical properties of a hRBC model can be obtained through the computational version of probing the hRBC. SMD simulation of a hRBC model can be used to determine if the model accurately reproduces the response of the actual hRBC.



## 2.5 Statistical Mechanics

### 2.5.1 Statistical Mechanics Background

Statistical mechanics (SM) uses probability theory to calculate the average bulk behavior of a system (classical thermodynamic properties) given some information about the nature and behavior of the constituents of the system. This is in contrast to MD, which explicitly models the positions and velocities of the particles in the system. The predictions given by SM can be very accurate, even though not all the microscopic details of the system are known. As mentioned previously in [SECTION 2.4](#), the complexity of MD simulations increases nonlinearly with the number of particles in the system, causing MD simulations to be computationally limited. SM can overcome this limitation by reducing the exact information of the system (positions and velocities of the particles) into a compact statistical form. The information in this subsection is primarily borrowed from a statistical mechanics text [50] and a computational biology text [51].

The critical concept of SM is that of microstates. A microstate is a particular configuration of the particles in the system. SM assumes that the particles in the system are indistinguishable, in accordance with the Heisenberg uncertainty principle. A macrostate is bulk state specified by a smaller number of parameters (usually classical thermodynamics properties such as temperature, pressure, volume, etc.). There is not a one-to-one correspondence between microstates and macrostates; in general, a large number of microstates correspond

to a single macrostate. Each microstate has an energy associated with it. Multiple microstates can have the same energy, even if their configurations are different.

The two fundamental postulates of SM are the equal a priori probability postulate and the ergodic hypothesis. The equal a priori probability postulate assumes that microstates with the same energy have the same probability of occurrence. The equal a priori probability postulate is a pure assumption, but it has not been contradicted by experimental evidence. The ergodic hypothesis assumes that the time average of a mechanical property of the system is equal the average value of the property over all of the microstates of the system, weighted by the probability of occurrence of the microstate. When the time average is sufficiently long (compared to the timescale of the system), the ergodic hypothesis allows for the time average to be replaced by the ensemble average.

As alluded to in [SECTION 2.3](#), a connection exists between CM and SM. The boundary where the transition from RVE to SVE occurs is known as the separation of scales. The equivalence of the RVE and the SVE at the separation of scales provides the connection between SM and CM. At the microscale, the stress-strain relationship of a material is governed by the Helmholtz free energy. The Helmholtz free energy is given by:

$$A = U - TS \quad (2.12)$$

where  $U$  is the internal energy of the system,  $T$  is the temperature, and  $S$  is the entropy. Systems tend towards configurations that minimize the Helmholtz free energy. This is why MD simulations need to be minimized from their initial configuration to their reference configuration before worthwhile simulations can

be performed. Equating the Helmholtz free energy across the separation of scales provides the connection between CM and SM.

### 2.5.2 Polymer Physics

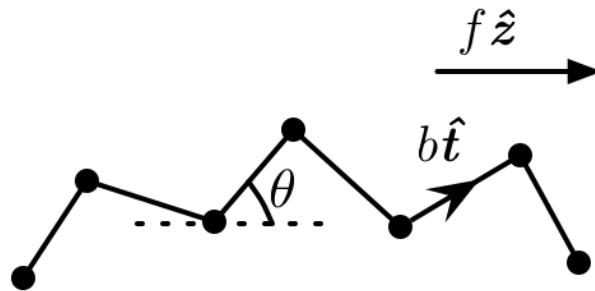
An understanding of SM is important in the context of this thesis because SM is used the field of polymer physics to model polymer chains. The worm-like chain (WLC) model of polymer elasticity is used to model the force-extension relationship of the spectrin links in the unified hRBC model. It is part of the two-body interaction potential in the force field of the hRBC model (described in [SUBSECTION 3.2.2](#)). A brief background on polymer physics, culminating with the WLC model, follows.

Any discussion of the history of polymer physics would be remiss to not mention Dr. Paul Flory. Considered by some to be the father of polymer physics, his contributions were critical to the development of the field, and many are still in use today. A collection of his lectures was published in 1953 [52], and this text became standard in the field of polymer physics.

A polymer is a chain composed of repeated subunits known as monomers. Polymers are ubiquitous in biology (all proteins, including spectrin and actin, are polymers). Two of the main quantities of interest in polymer physics are the molecular weight of the polymer chain and the end-to-end length of a polymer chain. The end-to-end length is different than the contour length of a polymer chain. The contour length is the sum of the lengths of the individual subunits in the chain. If the ends of the polymer chain were pulled in opposite directions until the chain was completely straight, the end-to-end length would be equal to

the contour length. Polymer chains are almost never found completely extended, like in the situation depicted previously. Instead, polymer chains in a solvent tend to form coils [48]. This is because the coil arrangement is more energetically favorable, as will be explained after the basic model of a polymer chain is introduced. The end-to-end length of a polymer chain can be affected by a variety of factors, including solvent conditions, temperature, and the application of a stretching force to the ends of the polymer chain. The stretching of polymer chains due to loading is most applicable to this thesis [53].

The most basic model of a polymer is the ideal chain model. In this formulation, the monomers of a polymer chain are modeled as identical, rigid links in a chain. The bonds between the links form the joints in the chain. The chain has  $N$  links, each of length  $b$ , giving the chain a contour length,  $L_m = Nb$ . This contour length is fixed, since neither the links nor the bonds between them can stretch (the ideal chain is inextensible). The chain is “freely-jointed”, which means that there is no resistance to inhibit links from rotating about a joint or bending at a joint with respect to one another (for this reason, the ideal chain model is sometimes referred to as the freely-jointed chain model). **FIGURE 2-9** is a depiction of the ideal chain model.



**FIGURE 2-9:** Depiction of the ideal chain model [54].

The end-to-end vector of the ideal chain model is given by:

$$\vec{R} = \sum_{i=1}^N \vec{r}_i \quad (2.13)$$

where  $\vec{r}_i$  is the vector position of the  $i$ -th link in the polymer chain (note, that  $\vec{r}$  corresponds to  $\hat{t}$  in [FIGURE 2-9](#)).

In the absence of additional information about the configuration of the polymer chain, the end-to-end length of the ideal chain can be modeled as a 3D random walk. In a random walk, the probability that a particle will be a particular displacement from its initial position after a certain number of steps can be described by a binomial distribution. The binomial distribution is a discrete probability distribution commonly used to model random sampling with replacement. In this context, it directly corresponds to a one-dimensional (1D) random walk where the number of steps is the number of links in the polymer chain and the length of each step is the length of the individual links in the polymer chain. A 3D random walk is equivalent to three 1D random walks in directions orthogonal to one another. As a result of the central limit theorem (more specifically, the de Moivre-Laplace theorem), the binomial distribution can be approximated by a Gaussian distribution for a larger number of steps in the random walk (links in the polymer chain) [48]. This distribution is given by:

$$P(\vec{R}, N) = \left(\frac{3}{2\pi Nb^2}\right)^{3/2} \exp\left(-\frac{3\vec{R}^2}{2Nb^2}\right) \quad (2.14)$$

where  $\vec{R}$  is the end-to-end vector of the polymer chain,  $N$  is the number of links in the chain, and  $b$  is the length of an individual link. This formulation is known as a Gaussian chain model. Note that, even though the contour length of the Gaussian chain model is  $Nb$ , extensions greater than the contour length are possible (although, they are highly improbable). Using the equal a priori probability postulate, this probability can be related to the number of possible microstates for each end-to-end length of the polymer chain:

$$\Omega(\vec{R}) \approx P(\vec{R}, N) \quad (2.15)$$

Note, this relation is approximate, since determining the actual form for the number of possible microstates requires quantum mechanical calculations. The entropy of the Gaussian chain is given by using Boltzmann's entropy equation:

$$S = k_B \ln(\Omega(\vec{R})) \quad (2.16)$$

The Helmholtz free energy can be calculated using [EQUATION 2.12](#). Taking the negative of the derivative of the Helmholtz free energy with respect to  $\vec{R}$  (and assuming that the internal energy is constant since the polymer chain is inextensible) yields a force-extension relationship for the Gaussian chain model:

$$\vec{F} = -\frac{\partial A}{\partial \vec{R}} = -\frac{3k_B T}{Nb^2} \vec{R} \quad (2.17)$$

Somewhat surprisingly, this relationship is linear (Hooke's law). A number of serious assumptions were made in this derivation, so it is not directly applicable for use in simulations. One of these assumptions is that the subunits of the

polymer can overlap one another. Because of this assumption, the ideal chain model does not display any excluded volume effects. Polymer chains that do not exhibit excluded volume effects are often known as phantom chains. Sometimes, a self-avoiding random walk is used in the derivation of the Gaussian chain model to account for some of these effects.

The force-extension relationship derived from the ideal chain model is an example of an entropic spring (because the free energy of the spring is a function of the entropy, not the internal energy). This change in entropy is directly related to a change in the number of microstates that the polymer chain can occupy. As the end-to-end length of the polymer chain is increased, the number of available microstates decreases, corresponding to a decrease in entropy. The concept of the entropic spring can also be observed in the WLC model.

As noted earlier, the Gaussian chain model is not very accurate in reproducing the force-extension behavior of polymer chains that is observed in experiments. An alternative derivation of the force-extension relationship for a polymer chain, known as the WLC model, represents an attempt to better reproduce these behaviors. The WLC model is applicable to “stiff” polymers. It is the continuous analogue to the discrete ideal chain model [55]. The WLC model was originally developed with the intent of modeling the force-extension relationship of DNA [56]. Since its development, the usage of the WLC model has been expanded to describe the behaviors of other polymers. **FIGURE 2-10** is a depiction of the WLC model.

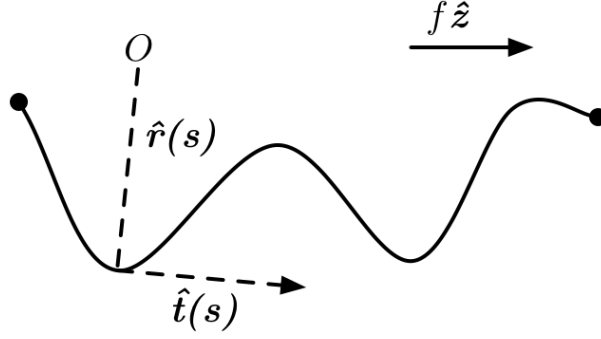


FIGURE 2-10: Depiction of the WLC model [54].

The WLC model envisions the polymer chain as a continuously flexible, isotropic rod. The derivation of the WLC produces an expression for the free energy of the polymer chain that depends on the bending of the chain. This equation is not differentiable, however, so an exact expression for the force-extension relationship does not exist. The force-extension relationship for the WLC model is approximated by an interpolation using the free energy expression and experimental results from DNA stretching tests as reference. The WLC model force-extension relationship is given by:

$$F = \frac{k_B T}{L_p} \left( \frac{1}{4(1-L/L_m)^2} - \frac{1}{4} + \frac{L}{L_m} \right) \quad (2.18)$$

where  $L_p$  is a quantity known as the persistence length. The persistence length is a measure of the stiffness of a polymer chain (a longer persistence length corresponds to a stiffer polymer). Mathematically, the persistence length is defined as the length where the polymer chain becomes uncorrelated with itself. Practically, the persistence length is often taken to be twice the length of a single link in the equivalent ideal chain ( $L_p = 2b$ ). The stiffness of the WLC model (derivative of force with respect to extension) about zero extension is given by:



$$K_{WLC}(L = 0) = \frac{3k_B T}{2L_m L_p} \quad (2.19)$$

An equivalence between the WLC model and the Gaussian chain model in the limit of small extensions can be obtained by comparing [EQUATION 2.19](#) to the coefficient in [EQUATION 2.17](#) (the spring constant) and using the relation that  $L_m = Nb$ . This completes the subsection on polymer physics. The WLC model will be revisited in [SUBSECTION 3.2.2](#).

## 2.6 Spring Networks and Percolation Theory

A spring network is a model of a system where the mass of the system is concentrated at specific points in space and springs with a defined stiffness connect the point masses. A spring network is a type of generalization of Hooke's law to two or three dimensions. The spring network formulation of an object is similar to a finite element formulation that can be utilized to solve statics problems. In general, spring networks are conservative and energy minimization methods are commonly employed in order to solve problems relating to the spring network. The exact connection between a 2D spring network and the hRBC model (hinted at in [SECTION 2.4](#)) will be fully detailed in [SUBSECTION 3.2.2](#).

The overall response of the network as a function of its parameters and geometry has been of great interest to researchers for years [57, 58]. There has been no shortage of attempts to model disordered spring networks as well [59, 60, 61, 62, 63]. Some spring network models replace the commonly used Hookean springs with other types of springs [64, 65, 66]. Some aspects of disordered models and the alternative spring models can be applied to biological materials [67, 68, 69, 70], and the cytoskeleton of the hRBC specifically [22, 23, 71, 72].

The spring network representation of the cytoskeleton is an attempt to model the micromechanics of the hRBC membrane.

A common theme in many of the disordered spring network models is the usage of mean field theory (MFT). The disordered nature of the spring network can be extremely intricate, with varying numbers of disconnected links in numerous locations in the network. MFT allows for a connection to be made between the connectivity of the spring network and the bulk properties of the spring network by allowing the complex interactions to be approximated by an average effect. This reduces the number of degrees of freedom of the problem, making it easier to gain insight from the analysis of the spring network. The accuracy of the approximated average effect varies, but in general the approximation is better for networks with a large number of springs and a high level of disorder (disconnected links are randomly distributed). In the context of obtaining bulk elastic properties from a discrete spring network, MFT represents a type of homogenization of the spring network. Percolation theory is commonly used as part of the MFT to describe these disordered spring networks [73, 74, 75, 76].

Percolation theory makes use of probability theory to predict the formation of long-range connectivity in random lattice systems. The theory gets its name from the flow of fluids through porous media, but it has been employed to help solve multiple other problems, the most common of which is the problem of conduction on a network of resistors. Most important to the problem at hand is the extension of percolation theory to the modeling of spring networks and

composite materials. Percolation theory can be generalized into two basic types: bond percolation and site percolation. In bond percolation, vacancies can form in the links that make up the lattice. In site percolation, vacancies can form in the vertices that make up the lattice. The results from bond percolation and site percolation are similar, but not identical. Bond percolation is of greater interest for the purposes of modeling spring networks.

In the case of bond percolation of a spring network, links are occupied with probability  $p$ , and vacant with probability  $1 - p$ . The value of  $p$  is the connectivity of the network. There are two critical values of the connectivity of the network: the connectivity percolation threshold ( $p_c$ ) and the rigidity percolation threshold ( $p_r$ ). The connectivity percolation threshold is the more commonly discussed of the two, and it is the value of the connectivity at which no continuous path of links exists from one side of the network to the other. The connectivity percolation threshold is commonly used when discussing resistor networks, where the global conductivity of the network is of interest. The rigidity percolation threshold is specific to the case of a spring network, and it is the value of the connectivity at which force can no longer be transmitted across the network due to an insufficient number of links. These are both mean field quantities, which means that they are averaged over the entire network surface. Even when the connectivity of the network is near one of these thresholds, the local behavior at a specific point in the network may not conform to the predictions obtained through percolation theory.

Both percolation thresholds are functions of the geometry of the network only, not the properties of the springs in the network, and can be calculated analytically for some lattice geometries. Using graph theory and probability theory arguments, the connectivity percolation threshold for the triangular lattice is calculated in [77], and is shown to be the solution of:

$$1 - 3p + p^3 = 0 \quad (2.20)$$

which has only one root between 0 and 1:

$$p_c = 2 \sin\left(\frac{\pi}{18}\right) \approx 0.347296 \quad (2.21)$$

Often, this value is approximated as:

$$p_c = \frac{1}{3} \quad (2.22)$$

Using effective-medium estimates and making a constraints argument, the rigidity percolation threshold is estimated in [74] by calculating the number of zero-frequency modes (floppy modes), which is given by the number of degrees of freedom minus the number of constraints:

$$f \approx \frac{Nd - \frac{1}{2}zNp}{Nd} \quad (2.23)$$

where  $N$  is the number of links in the network,  $d$  is the dimensionality of the network ( $d = 2$  for the case of a 2D spring network), and  $z$  is the degree of the network ( $z = 6$  for the case of a perfectly triangular network). **EQUATION 2.23** is an approximation because some links in the network are redundant (they do not provide an additional constraint on the number of degrees of freedom) [48]. Solving **EQUATION 2.23** for  $p$  for the case where the number of zero-frequency modes is zero yields:

$$p_r = \frac{2d}{z} \quad (2.24)$$

In the case of the 2D triangular network, the rigidity percolation threshold is:

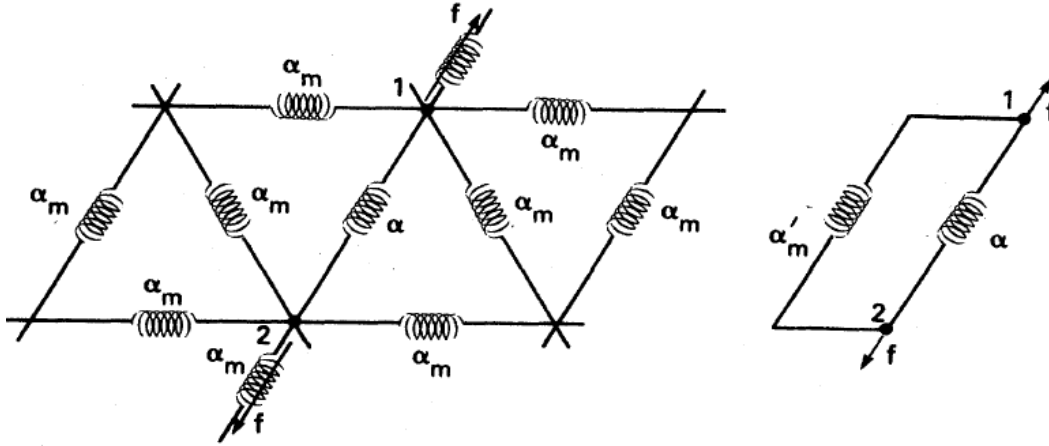
$$p_r = \frac{2}{3} \quad (2.25)$$

It is apparent from [EQUATION 2.24](#) and [EQUATION 2.25](#) that:

$$p_r > p_c \quad (2.26)$$

This relationship implies that rigidity in a spring network requires more intact links than connectivity. This is significant because the local behavior of the cytoskeleton is closely related to the connectivity percolation threshold, while the global behavior (macroscopic elastic properties) is closely related to the rigidity percolation threshold [74].

An argument is presented by Feng et al. [74] (and reproduced by Zhang et al. [78]) that makes a connection the effective spring constant and the connectivity of a randomly diluted spring network using the rigidity percolation threshold. A graphical representation of the concept behind this formulation is shown in [FIGURE 2-11](#).



**FIGURE 2-11:** Effective medium formulation of diluted spring network used by Feng et al. [74].

The network is populated by springs with identical spring constants ( $\alpha_m$ ), as shown in the left portion of **FIGURE 2-11**. When examining the force-extension interaction between sites 1 and 2, the entire spring network can be reduced to a spring with an effective spring constant ( $\alpha'_m$ ) in parallel with the spring between sites 1 and 2, as shown in the right portion of **FIGURE 2-11**. The effective spring constant accounts for the contributions from the rest of the network towards the force-extension behavior of the spring between sites 1 and 2. If the spring between 1 and 2 is disconnected (its spring constant is zero), the force-extension interaction between sites 1 and 2 is governed by the contribution from the rest of the network only. Then, by using an effective-medium assumption to relate the fluctuations in the value of the extension between sites 1 and 2 to its time-average value, a mean spring constant for the spring between sites 1 and 2 as a function of the connectivity can be obtained. This relationship is given by:

$$\frac{\alpha}{\alpha_m} = \frac{p-p_{cen}}{1-p_{cen}} \quad (2.27)$$

where  $\alpha$  is the mean spring constant between sites 1 and 2, and  $p_{cen}$  is the rigidity percolation threshold in their formulation.

A relationship between the spring constants and elastic moduli was presented by Boal [48]. For the shear modulus and the bulk modulus, these relationships are:

$$\mu = \frac{\sqrt{3}}{4} k_{sp}$$

$$K_A = 2\mu \quad (2.28)$$

where  $\mu$  is the shear modulus,  $K_A$  is the bulk modulus, and  $k_{sp}$  is the spring constant of the springs in the network. Combining [EQUATION 2.27](#) with [EQUATION 2.28](#) provides a relationship for the elastic moduli of a diluted spring network as a function of the connectivity of the network (used in [SUBSECTION 3.4.1](#)).

An understanding of percolation theory is important in the context of hRBC modeling because the cytoskeleton can be formulated as a disordered triangular spring network, as hinted at earlier in [SUBSECTION 2.1.2](#). In this network, the probability of an individual spring being connected is  $p$ . As the connectivity of the spring network changes (the local “character”), it follows logically that the response of the network as a whole will change also. The relationship between connectivity and macroscopic elastic properties that is used in this thesis is presented in [SECTION 3.4](#).

## 2.7 Triangulation

A triangulation is the creation of mesh from a set of points distributed on the surface of an object. The mesh consists of a set of 2D, triangular elements (hence, the name triangulation). The points in the triangulation are called vertices (or nodes), and the elements are sometimes referred to as triangles. Neighboring elements share vertices and edges (there is no empty, unaccounted for space). The concept behind triangulation is the representation of a smooth, continuous surface with a set of discrete elements. Computer simulations can be performed utilizing this representation (similar to the idea behind the finite element method) [79]. In a triangulation, each node is assigned a unique number (between one and the number of nodes in the triangulation) that is used as an identifier. The set of elements and the set of links in the triangulation are both defined in terms of this unique identifier (as opposed to being in terms of the coordinates of the node positions). The concept is similar to the use of indices to identify components in a vector or matrix.

Triangulation is important in the context of MD simulations of the hRBC because the hRBC is represented by a set of points in space, and the connectivity of these points needs to be determined so that the force-field can be correctly applied. The discrete, inhomogeneous structure of the hRBC (as mentioned in [SUBSECTION 2.1.2](#)) actually lends itself to discretization. In the process of the triangulation, the initial positions of the points are also defined. These initial positions are used in the MD simulation as initial conditions for the first step of the numerical integration of the equations of motion.



The triangulation method required for this thesis is a bit unusual and was difficult to find. Most triangulations are of 2D surfaces that lie in a plane. Most “triangulations” of 3D objects actually mesh the volume itself (forming tetrahedrons instead of triangles), instead of creating elements that represent just the surface of the object. In order to properly triangulate the hRBC, a surface meshing routine that can operate on closed surfaces is needed.

Neglecting the edges of the triangulation (assuming that the triangulation is infinite in a 2D plane), relationships between the number of vertices (nodes/points), the number of elements (triangles), and the number of edges (links/sides) in the network can be expressed:

$$\begin{aligned} N_t &= 2N_v \\ N_s &= \frac{3}{2}N_t \end{aligned} \tag{2.29}$$

where  $N_v$  is the number of vertices,  $N_t$  is the number of triangles, and  $N_s$  is the number of edges. The first relationship can be determined through the observation that each vertex is shared among 6 elements and each element has 3 vertices. The second relationship can be determined through the observation that each element has three edges and each edge is shared between two elements.

Creating the point cloud and defining the connectivity between the points is the first step in creating the virtual version of the hRBC model that can be used in simulations. Then, analytical analysis can relate the parameters of the individual springs in the network to the bulk parameters of the spring network as a whole, under the assumption of a perfectly triangular network. The closer the connectivity of virtual model is to that of the perfectly triangular network (entirely

degree-6), the closer the behavior of the virtual model will be to the behavior predicted by the analytical analysis. Any deviation of the triangular elements in the virtual network from the ideal, equilateral shape will result in error in the simulation results. Therefore, it is important that the initial triangulation be of high-quality.

However, no closed triangulation is perfect, even with the best algorithm and the triangulation in its reference configuration and/or minimum energy state. These triangulations will have defects, which means that not all the vertices will be degree-6 [80]. Both Descartes' law of closure deficit and the Euler characteristic formula can be used to determine that a closed triangulation will have at least 12 defects. Because of this, a different relationship between the number of elements and the number of vertices exists:

$$N_t = 2N_v - 4 \quad (2.30)$$

The relationship between the number of edges and the number of elements is the same as above. The presence of these defects can result in unexpected stress concentrations in the hRBC model. As the number of nodes in the triangulation is increased, the effect that these defects have on the overall character of the triangulation is decreased.

There are a number of measures of triangulation quality. Since the hRBC model used in this thesis is borrowed from Fedosov et al. [44], the quality of the triangulation utilized for simulations is compared to the quality of their triangulations using the measures of triangulation quality presented in [44].

The two measures of triangulation quality used by Fedosov et al. are:

1. Distribution of the link (edge) lengths
2. Distribution of the vertex degrees (number of links in the vertex junction)

A smaller distribution of link length and a higher distribution of vertex degree corresponds to a higher quality triangulation. Their best average mesh quality was obtained from the “energy relaxation” method. The distribution of link length is given by:

$$d(l) = \frac{\sigma(l)}{\bar{l}} \quad (2.31)$$

where  $\sigma(l)$  is the standard deviation of link lengths and  $\bar{l}$  is the average link length.

In addition, another measure of triangulation quality from [81] is utilized in this thesis:

$$Q = \frac{(L_B + L_C - L_A)(L_C + L_A - L_B)(L_A + L_B - L_C)}{L_A L_B L_C} \quad (2.32)$$

where  $A$  is the area of the element in question and  $L_i$  is the length of each side ( $i = A, B, C$ ). Effectively, this quality measure is the ratio of the radius of the largest inscribed circle of the triangle to the radius of the smallest circumscribed circle.

Using this quality measure, an equilateral triangle has a quality  $Q = 1$ , while a degenerate triangle (zero area) has a quality  $Q = 0$ . The quality of the entire triangulation is the average of the individual element qualities over the entire triangulation. In general, a quality  $Q \geq 0.5$  indicates that the triangulation is of good quality.

## Chapter 3: Research Methods and Techniques

### 3.1 Triangulation of the hRBC

The triangulation used in this thesis is created using *distmeshsurface.m* (and its related functions) from “A Simple Mesh Generator in MATLAB” [81]. This triangulation function was chosen because it generates a high-quality, closed triangulation for the provided geometry in a reasonable amount of time. A brief description of the algorithm with respect to its usage in the hRBC simulation in this thesis follows. The specific usage of *distmeshsurface.m* and related functions is detailed after the description of the algorithm. A full description of the algorithm and its MATLAB® implementation can be found in [79, 81].

The concept behind the algorithm used in the *distmeshsurface.m* function is the analogy between a mesh and a truss structure. The nodes and links in the mesh correspond to the joints and members of the truss. The geometry of the surface is represented by a signed distance function that is negative inside the surface, zero on the boundary, and positive outside the surface. Initially, an implicit expression for the surface of the hRBC surface is given to the algorithm. The expression is given by:

$$0 = z \pm D_0 \sqrt{1 - \frac{4(x^2 + y^2)}{D_0^2}} \left[ a_0 + a_1 \frac{x^2 + y^2}{D_0^2} + a_2 \frac{(x^2 + y^2)^2}{D_0^4} \right]$$
$$x^2 + y^2 \leq \frac{D_0^2}{4} \quad (3.1)$$

where  $z$  is the height (in  $\mu\text{m}$ ) of the hRBC surface above the x-y plane,  $D_0$  is the major diameter (in  $\mu\text{m}$ ) of the hRBC in its unstressed, reference configuration,

and  $a_0$ ,  $a_1$ , and  $a_2$  are shape constants. Next, a 3D box that completely encompasses the shape given in the implicit expression is generated (bounding box). The box is then populated with points that are equally spaced (specified spacing) on a 3D grid. Points that lie outside a specified threshold distance of the surface are discarded. The remaining points ( $N_v$ ) are then triangulated using the Delaunay triangulation routine in MATLAB®. An iterative process is then used to enhance this initial triangulation. During this process, the node locations are optimized using a force-based smoothing procedure and the topology is updated according to the Delaunay triangulation routine. The force-based smoothing procedure adjusts the positions of the nodes in an attempt to make the links in the triangulation have equal lengths (which drives the shape of the individual elements towards equilateral triangles). After the adjustment of node positions, the positions are then projected back onto the boundary of the surface (as specified by the signed distance function). If there are significant changes in node positions, all the nodes are retriangulated. This whole process is repeated until the changes in node positions are less than a specified distance, and the vertex locations and the elemental connectivity are then output.

The triangulation is generated by running the *hRBC\_Triangulation.m* script. This script is a wrapper for the triangulation generation procedure. The implicit expression for the surface of the hRBC (and its associated shape constants), the locations of the corners of the bounding box, and the grid spacing for the points in the bounding box are specified in the *hRBC\_Triangulation.m* script. The *distmeshsurface.m* function is then called with these quantities as

inputs, as well as with the *huniform* argument. This argument sets the condition that the lengths of the links in the triangulation should be equal.

In the *distmeshsurface.m* function, parameters for the triangulation generation are specified (termination condition for the procedure, maximum nodal displacement before a retriangulation is required to update the topology, the “internal pressure” that drives the node location optimization, the time step length for the force-based smoothing procedure, and the geometric tolerance used in the numerical differentiation of the signed distance function). Next, the built-in MATLAB® function *ndgrid* is used to populate the bounding box with points. Next, the built-in MATLAB® function *isosurface* is used to select the points from the grid that are inside the surface (interior points) and within a specified distance of surface. The *isosurface* function also outputs a set of elements that form the initial triangulation of the remaining points. The remaining points now form the set of nodes in the hRBC network. The *mkt2t.m* function is then used to compute the connectivity of the elements from the output of the *isosurface* function.

The newly initialized mesh is now optimized iteratively through triangulation refinement and the use of the force-based smoothing procedure, as discussed above. The triangulation is updated every time the change in a node’s position is greater than the specified tolerance using the *trisurfupd.cpp* function. This function (called through the use of a –MEX wrapper) updates the triangulation by flipping the edges of neighboring elements in the triangulation and/or adding new elements as necessary. It is coded in C++ to improve the computational efficiency of the *distmeshsurface.m* function (since the

retriangulation is often the most computationally expensive portion of the entire triangulation process). The triangulation progress is then graphically output to the screen. After the triangulation is updated, the positions of the nodes are updated using the force-based smoothing procedure. This procedure calculates the lengths of all the links in the triangulation and then then the forces that are required to obtain a triangulation with uniform link lengths. These forces are then applied to the appropriate nodes in the triangulation, and the positions are updated. The magnitude of the force is proportional to the specified “internal pressure”. After the nodal positions are updated by the force-based smoothing procedure, each node is projected back onto the boundary of the surface using the signed distance function. The gradient of the signed distance function is numerically calculated and then used to find a vector that passes through the node and is normal to the surface. The node is then shifted along the vector in the direction of the surface. The magnitude of the shift is such that the signed distance function will be zero after the node is shifted (meaning that the node is back on the surface). This completes one iteration of the optimization. The optimization procedure is complete when the change in position for all nodes (after both the force-based smoothing procedure and the projection of the nodes back onto the boundary of the surface are performed) is less than the tolerance specified at the beginning of the *distmeshsurface.m* function. The *distmeshsurface.m* function outputs the location of the nodes ( $N_v$ -by-3 matrix that contains the x, y, and z coordinates of the node locations) and the set of elements in the triangulation ( $N_t$ -by-3 matrix that contains the three nodes that form each element).

The output from the *distmeshsurface.m* function is then used as input to create a TriRep class in the *hRBC\_Triangulation.m* script file. The set of links ( $N_s$ -by-2 matrix that specifies which nodes are connected to one another) is calculated from the TriRep class using the *edges* function. The set of links and the set of elements are then input into the *f\_connectivity.m* function in order to determine which elements in the triangulation are neighbors. Neighboring elements share one edge, and the faces of two elements can bend with respect to one another about this edge (this is the bending that is defined in [SUBSECTION 3.2.2](#)). The *f\_connectivity.m* function then uses the list of neighboring triangles to determine the set of bending points ( $N_s$ -by-4 matrix that contains the four nodes that can be used to describe neighboring elements). The node locations, set of elements, set of links, and set of bending points as well as the number of nodes, number of elements, and number of links are saved in *hRBC\_Triangulation.mat* for use in simulations of the hRBC.

In the matrices that contain the set of elements, the set of links, and the set of bending points, the relevant nodes are stored using their unique identifiers. Because of this, the order of the nodes in the matrices is extremely important. The code used to simulate the hRBC assumes that the nodes are in the correct order. If the order of the nodes is incorrect in the matrix that contains the set of elements, the normal vectors from the elements will be calculated incorrectly. If the order of the nodes is incorrect in the set of bending points, the link about which the two elements are bending with respect to each other will be incorrect.



This is obviously undesirable, so the *connectivity.m* function contains checks to ensure that the order of the nodes is correct in each matrix.

This completes the triangulation used for the hRBC. Next, a coarse-grained MD force field is defined and interactions between the nodes are defined so as to reproduce the morphology and mechanical behavior of the hRBC.

## 3.2 Mechanical Model of the Human Red Blood Cell

### 3.2.1 Past Models of the Human Red Blood Cell

Past models of the hRBC treated the cell as a continuous elastic body or a thin elastic shell enclosing an incompressible fluid (the cytoplasm) [32, 33, 82].

If the hRBC membrane is modeled as an incompressible effective continuum, the in-plane shear stress is given by:

$$T_s = \mu(\varepsilon_1 - \varepsilon_2) \quad (3.2)$$

where  $\mu$  is the shear modulus and  $\varepsilon$  are the in-plane strains. The incompressibility condition is enforced by requiring:

$$(\varepsilon_1 + 1)(\varepsilon_2 + 1) = 1 \quad (3.3)$$

If the hRBC membrane is modeled using a hyperelastic effective material model using the first order neo-Hookean formulation, the strain energy potential function is given by:

$$U = \frac{G_0}{2} [(\varepsilon_1 + 1)^2 + (\varepsilon_2 + 1)^2 + (\varepsilon_3 + 1)^2 - 3] \quad (3.4)$$

where  $G_0$  is the initial bulk shear modulus. If the membrane is assumed to be incompressible, the incompressibility condition is enforced by requiring:

$$(\varepsilon_1 + 1)(\varepsilon_2 + 1)(\varepsilon_3 + 1) = 1$$

The in-plane shear stress is given by:

$$T_s = \frac{G_0 h_0}{2} ((\varepsilon_1 + 1)^{1.5} - (\varepsilon_1 + 1)^{-1.5}) \quad (3.5)$$

where  $h_0$  is the initial membrane thickness. The instantaneous shear modulus is given by:

$$\mu = \frac{1}{2} \frac{\partial T_s}{\partial \gamma_s} = \frac{3G_0 h_0 ((\varepsilon_1 + 1)^{0.5} - (\varepsilon_1 + 1)^{-2.5})}{4((\varepsilon_1 + 1) + (\varepsilon_1 + 1)^{-3})}$$

Note that, in the hyperelastic formulation, the shear modulus is not constant. Instead, it rapidly decreases from its initial value of  $\mu_0 = 0.75 G_0 h_0$ , and then asymptotically approaches a relatively smaller value  $\mu_1$ .

While having the advantages of simplicity and a clear connection with CM principles, simulations making use of the above continuum models were unable to reproduce the behaviors of hRBC's observed in experiments. In addition, "Fully continuum (fluid and solid) modeling often suffers from difficulties in coupling nonlinear solid motions and fluid flow without excessive computational expense" [44]. Because of these issues, recent hRBC modeling efforts have taken an alternate approach. Instead of treating the hRBC as a continuous elastic body, the surface of the hRBC (the cytoskeleton and the lipid bilayer) are modeled as discrete elements (somewhat similar to a FEM formulation). This approach allows for better coupling between the fluid and solid domains, as well as the possibility of better reproducing hRBC morphologies in the future. The hRBC model used in this thesis is described next in [SUBSECTION 3.2.2](#).

It is important to note that, in the majority of these discrete models of the hRBC, the cytoskeleton and the lipid bilayer are modeled as a single component. As discussed in [SUBSECTION 2.1.2](#), this is not actually correct, because the cytoskeleton and the lipid bilayer are free to move relative to each other at points other than their attachment points (and there is even some drift at the attachment points due to lateral diffusion in the lipid bilayer of the proteins the cytoskeleton is anchored to). That being said, the bilayer-cytoskeleton couple hypothesis

seems to be a fair approximation in reproducing the mechanical behavior of the hRBC [83, 84].

Some other important efforts of discrete modeling of the hRBC were not mentioned above. The efforts by Boey et al. [85] provided information regarding the behavior of single polymer chains in the cytoskeletal network as well as the cytoskeletal network as a whole. These efforts also made micropipette aspiration simulations of hRBC models possible [86]. The efforts of Li et al. [87] produced a model of the hRBC membrane that was used to determine the individual contributions from the cytoskeleton and the lipid bilayer to the overall response of the hRBC under loading. The efforts of Jiang et al. [88] provided information on the relationship between the stiffness of the individual spectrin fibers and the deformability of the cell as a whole. The efforts by Li et al. [80, 89] produced simulations of a coarse-grained hRBC model with varying number of vertices and network connectivities. The efforts by Peng et al. [90, 91] studied models of hRBC components at different length scales and ultimately produced a methodology for simulating a hRBC model in Stokes flow.

### 3.2.2 Current Model of the Human Red Blood Cell

The hRBC model that is used in this thesis was initially developed by Discher et al. [86], modified by Pivkin et al. [92], and then further improved upon by Fedosov et al. [44, 93, 94]. It will be referred to as the CG-hRBC (Coarse-Grained – human Red Blood Cell) model in this thesis. The CG-hRBC model was chosen because, in addition to having been used in previous hRBC modeling/simulation efforts [1], it also corrects some discrepancies in a previous analysis of the connection between model parameters and mechanical properties of the hRBC (performed by Dao et al. [95]).

The basis of the formulation of the CG-hRBC model is the treatment of the cytoskeleton of the hRBC as an infinite, 2D, triangular spring network with the actin nodes as point masses at the junctions of the springs. This approximation is valid locally for a high enough node density (large number of nodes in the model) where the curvature of the hRBC surface vanishes. Linear elastic response and affine deformations are also assumed. A brief description of the model follows.

The free energy of the CG-hRBC model is given by the expression:

$$U_{TOT} = U_{in-plane} + U_{area} + U_{volume} + U_{bending} \quad (3.6)$$

This is the basis of the force field for the MD simulation. The first term in the free energy expression accounts for the two body interaction between topologically connected nodes. It takes the form:

$$\begin{aligned}
U_{in-plane} &= \sum_{j \in 1 \dots N_s} (U_{POW} - U_{WLC}) \\
&= \sum_{j \in 1 \dots N_s} \left( \frac{k_p}{(1-m)L^{m-1}} - \frac{k_B T L_m}{4L_p} \frac{3x^2 - 2x^3}{1-x} \right)
\end{aligned} \tag{3.7}$$

where  $N_s$  is the number of links in the triangulation,  $k_p$  is the power law spring parameter,  $m$  is the exponent in the power law,  $L$  is the length of link extension,  $k_B$  is Boltzmann's constant,  $T$  is the temperature,  $L_m$  is the contour length of the link (maximum length of spectrin extension),  $L_p$  is the persistence length of the link, and  $x = L/L_m$ . The power law portion (POW) accounts for the steric repulsion between topologically connected nodes as well as the energy required to severely bend a local portion of the cell membrane. The worm-like chain portion (WLC) accounts for the attraction between topologically connected nodes due to the spectrin links.

The second term in the free energy expression accounts for the three body interaction between topologically connected nodes (local area conservation) as well as a global area conservation. It has the form:

$$\begin{aligned}
U_{area} &= U_{global\ area} + U_{local\ area} \\
&= \frac{k_a (A - A_0^{tot})^2}{2A_0^{tot}} + \sum_{j \in 1 \dots N_t} \frac{k_d (A_j - A_0)^2}{2A_0}
\end{aligned} \tag{3.8}$$

where  $k_a$  is the global area conservation constraint,  $A$  is the total surface area of the hRBC,  $A_0^{tot}$  is the equilibrium total surface area of the hRBC,  $N_t$  is the total number of elements (triangles) in the triangulation,  $k_d$  is the local area conservation constraint,  $A_j$  is the area of each individual element in the

triangulation, and  $A_0$  is the equilibrium area of each individual element in the triangulation.

The third term in the free energy expression accounts for global volume conservation. It has the form:

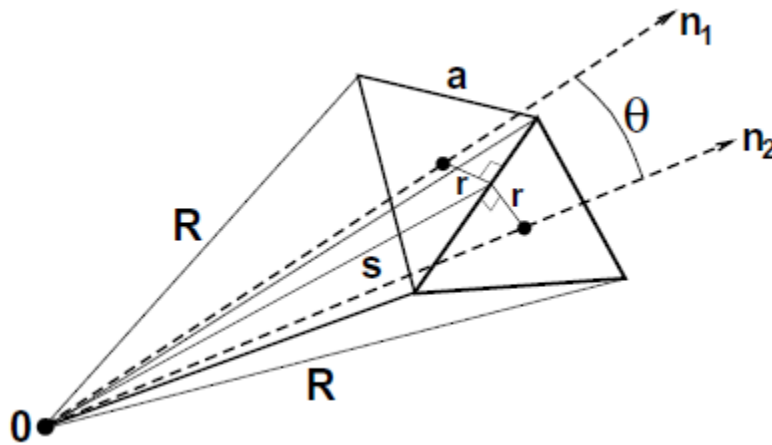
$$U_{volume} = \frac{k_v(V - V_0^{tot})^2}{2V_0^{tot}} \quad (3.9)$$

where  $k_v$  is the global volume conservation constraint,  $V$  is the total volume of the hRBC, and  $V_0^{tot}$  is the equilibrium total volume.

The fourth term in the free energy expression accounts for bending between adjacent elements in the triangulation. It accounts for the four body interaction between the nodes in neighboring elements. It has the form:

$$U_{bending} = \sum_{j \in 1 \dots N_s} k_b [1 - \cos(\theta_j - \theta_0)] \quad (3.10)$$

where  $N_s$  again is the number of links in the triangulation,  $k_b$  is the bending constraint,  $\theta_j$  is the angle between the normal vectors of adjacent elements (illustrated in **FIGURE 3-1**), and  $\theta_0$  is the equilibrium angle between the normal vectors of adjacent elements.



**FIGURE 3-1:** Depiction of bending angle between the normal vectors ( $n_1$  and  $n_2$ ) of adjacent elements [93]

This force field, as described above, is a conservative force field, meaning that the total energy of the system is conserved. It is also independent of time. The inclusion of viscoelastic terms (designed to account for the viscosity of the cell membrane, the cytoplasm, and the blood plasma) can transform the force field into a non-conservative one. By taking the gradient with respect to displacement, the forces acting on the set of particles under the influence of these interaction potentials can be derived. The exact forms of the force expressions can be found in Appendix A of [93].

Now that the functional forms of the terms in the free energy expression (EQUATION 3.6) have been chosen, the parameters in those expressions have to be related to physical properties of the hRBC that have been measured in experiments. The shear modulus  $\mu$  and the bulk modulus  $K$  have been measured experimentally, and found to be  $\mu \approx 6.3\text{e-}6$  N/m and  $K \approx 3.3\text{e-}4$  N/m, respectively (Note that these are 2D moduli of the membrane, therefore the units differ from the pressure units used with the usual, 3D). A connection between the parameters and the properties can be derived by using the virial stress (from the virial theorem) [95].

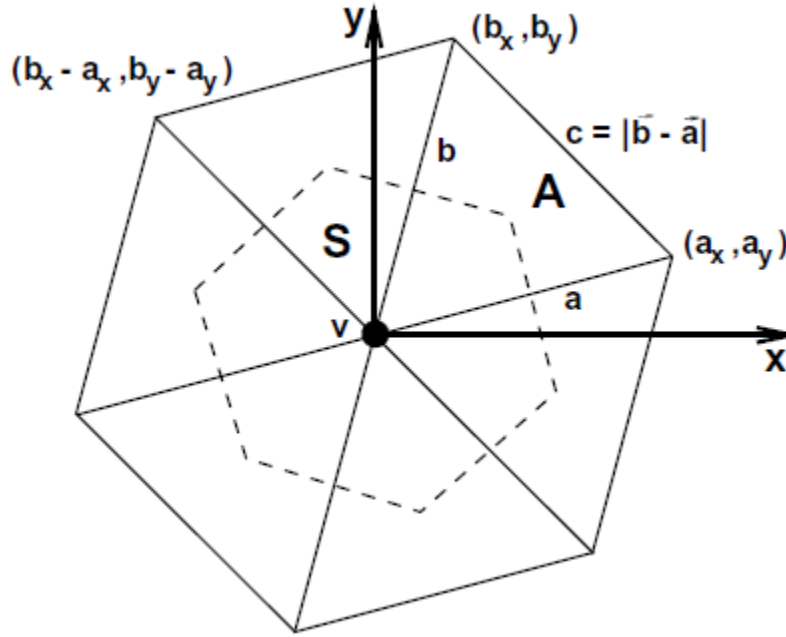
The virial stress is used to determine the macroscopic (continuum) stress in molecular dynamics simulations. It is the microscopic (discrete) analog to the Cauchy stress. The 2D version of the virial stress used in this derivation takes the form:



$$\tau_{\alpha\beta} = -\frac{1}{2A} \left[ \frac{f(a)}{a} a_\alpha a_\beta + \frac{f(b)}{b} b_\alpha b_\beta + \frac{f(c)}{c} (b_\alpha - a_\alpha)(b_\beta - a_\beta) \right] - \frac{(k_a + k_d)(A_0 - A)}{A_0} \delta_{\alpha\beta} \quad (3.11)$$

where  $f(\cdot)$  is the in-plane force, and the indices  $\alpha$  and  $\beta$  can be either  $x$  or  $y$ . The link lengths  $a$ ,  $b$ ,  $c$  are indicated on the representative area element (RAE) in

**FIGURE 3-2.**



**FIGURE 3-2:** Representative area element (RAE) from cytoskeletal network [93]

Note that  $\tau$  acts in the plane of the network (2D stress), and it only has  $x$  and  $y$  components. As an aside, the RAE in **FIGURE 3-2** is the 2D equivalent of the RVE introduced in **SECTION 2.3**. The surface area of the RAE is:

$$A = \frac{1}{2} |a_x b_y - a_y b_x| \quad (3.12)$$

The shear modulus of the RAE can be determined by taking the derivative of the shear stress ( $\alpha = x$ ,  $\beta = y$ ) about the reference configuration when an infinitesimal shear strain is applied [95]. The shear modulus is:

$$\mu_0^{WLC-POW} = \frac{\sqrt{3}k_B T}{4L_p L_m x_0} \left( \frac{x_0}{2(1-x_0)^3} - \frac{1}{4(1-x_0)^2} + \frac{1}{4} \right) + \frac{\sqrt{3}k_p(m+1)}{4L_0^{m+1}} \quad (3.13)$$

where  $x_0 = L_0/L_m$ , and  $L_0$  is the equilibrium length of the links. It is important to note that neither the area conservation constraints ( $k_a$  and  $k_d$ ) nor the volume conservation constraint ( $k_v$ ) appear in the shear modulus.

A major assumption made in the above derivation of the bulk elastic properties of the model network is that the deformations are affine, which means deformations to the network are independent of scale (local deformations are linearly proportional to global deformations). The affine deformation assumption is commonly made in CM analysis (especially with the use of RVEs/RAEs).

The bulk modulus of the representative element can be determined by taking the derivative of an area expansion stress ( $\alpha = x, \beta = x$  or  $\alpha = y, \beta = y$ ) about the reference configuration when an infinitesimal area expansion strain is applied [44]. The bulk modulus is:

$$K = 2\mu_0^{WLC-POW} + k_a + k_d \quad (3.14)$$

The shear and bulk moduli can be related to Young's modulus ( $Y$ ) and Poisson's ratio ( $\nu$ ) through the expressions:

$$Y = \frac{4K\mu_0}{K+\mu_0} \quad (3.15)$$

$$\nu = \frac{K-\mu_0}{K+\mu_0} \quad (3.16)$$

Note that the above expressions differ from the elastic moduli relationships commonly found in elasticity analysis ([EQUATION 2.4](#) and [EQUATION 2.5](#)). This discrepancy is because the first set of relationships are for 2D moduli, while the

second set of relationships is for 3D moduli. Relationships between the 2D moduli and the 3D moduli can be derived under the assumption that the 2D relationships between the 2D moduli were derived from the 3D relationships between the 3D moduli for the conditions of either plane stress or plane strain [96, 57]. Under the assumption of plane stress, the relationships between the 2D moduli and the 3D moduli are:

$$Y_2 = Y_3 \quad (3.17)$$

$$\nu_2 = \nu_3 \quad (3.18)$$

where the subscripts 2 and 3 refer to 2D and 3D, respectively. It is interesting to note that, under the assumption of plane strain, the 2D and 3D moduli are equal in magnitude (but they had different units). Under the assumption of plane strain, the relationships between the 2D moduli and the 3D moduli are:

$$Y_2 = \frac{Y_3}{(1-\nu_3^2)} \quad (3.19)$$

$$\nu_2 = \frac{\nu_3}{(1-\nu_3)} \quad (3.20)$$

where the subscripts again refer to the dimensionality of the moduli. The assumption of plane strain is made in this thesis.

For an incompressible, 2D material,  $\nu = 1$  (this differs from an incompressible 3D material, where  $\nu = 0.5$ ). For the incompressible membrane assumption to be true,  $K \gg \mu_0$ . This in turn implies that  $k_a + k_d \gg \mu_0$ .

Setting the in-plane force to zero when the link is at its equilibrium length provides the final relationship between parameters:

$$\frac{k_p}{L_0^m} = \frac{k_B T}{L_p} \left( \frac{1}{4(1-x_0)^2} - \frac{1}{4} + x_0 \right) \quad (3.21)$$

Qualitatively, the response of a network with the force field described above is isotropic for small deformation, but anisotropic for large deformations (even if the deformation is area-preserving). The lipid bilayer behaves as if it were a 2D fluid. The cytoskeleton accounts for all of the deviatoric resistance, while the lipid bilayer accounts for nearly all of the hydrostatic resistance.

In the CG-hRBC model, dissipative and random forces are also included as part of the force field so that dissipative particle dynamics (DPD) [97] simulations of the model can be performed. The contributions from these forces are calculated according to the fluid particle model [98]. As discussed in [SUBSECTION 4.5.6](#), it is the belief of the author of this thesis that the fluid particle model is not applicable in coarse-grained MD simulations of the hRBC for both theoretical and computational reasons. Instead, an alternate formulation to account for the viscosity (described in [SUBSECTION 3.2.3](#)) is used.

### 3.2.3 Modifications to CG-hRBC Model

The worm-like chain model does not exhibit the same behavior as an actual spectrin filament at large extensions. The worm-like chain model is inextensible, like the ideal chain model. As a result of the mathematical formulation of the worm-like chain model, the restoring force of the link goes to infinity as the length of the stretched link approaches its contour length. This is not physically accurate, as there should be some point where the force in the links reaches a critical value and the link yields or fractures. The rupture of spectrin has been modeled in simulation [99, 100].

The idea of implementing yielding/fracturing links was inspired the cytoskeletal dynamics model proposed by Li et al. [101]. Seemingly paradoxical behavior was observed between classical membrane theory [102] and simulations used to determine the equilibrium shape of the hRBC. Calculations using classical membrane theory [103, 104, 35, 105] required that only the bending energy of the membrane needs to be minimized in order to reproduce the biconcave equilibrium shape of the hRBC. The minimization of only the bending energy in simulations of hRBC models, however, did not result in the biconcave shape of the hRBC at equilibrium. The in-plane energy is several orders of magnitude greater than the bending energy, and the biconcave shape can only be reached if both energy terms are minimized. Even with minimization of the in-plane energy, stress concentrations can still result in the network due to an imperfect triangulation and/or the defects present in any closed surface triangulation [80]. Li et al. [101] propose a hypothesis that, “In an ideal limit, for

any RBC shape, the cytoskeleton always undergoes remodeling in topological connectivity at a certain rate to relax its in-plane shear elastic energy to zero". To test this hypothesis, they propose a "minimal representation of the cytoskeleton geometry that can self-organize and dynamically evolve" and include "mechanisms for nonthermal energies such as the strain energy or specific biochemical energy to influence and regulate structural evolution". This dynamic evolution of the connectivity of the cytoskeletal network in order to eliminate stress concentrations is the underlying motivation in this thesis.

To test this hypothesis, Li et al. created a computation model of the hRBC cytoskeleton network, known as the cytoskeletal dynamics model. The spectrin links in the network are modeled as chains of 39 beads each (intended to represent the 22 triple-helical repeats in  $\alpha$ -spectrin and the 17 triple-helical repeats in the  $\beta$ -spectrin). The actin nodes in the network are also represented by beads. Beads that are topologically connected in the chains are linked by a harmonic potential with a nonzero equilibrium distance. The spectrin beads that are at the ends of the chains are linked to their respective actin beads with a Lennard-Jones (LJ) potential. The beads were arranged to form a hexagonal network, with the actin nodes forming the vertices of the network. Simulations using this network could be performed. Some simulations involved stretching of the network in order to obtain its elastic properties, while other simulations attempted to imitate the effects of ATP on the cytoskeletal network through the addition of kinetic energy to random beads in the network. The use of ATP inspired further investigation

into its effects on the cytoskeleton, while the bead structure of the network motivated a modification of the CG-hRBC model.

This modification was made to the in-plane interaction energy expression used in the model in order to better reproduce the behavior of a spectrin filament at large extensions. This modification involved changing the expression to a piecewise function, where the force-extension relationship is still given by the WLC-POW expression in the low extension regime, but is now given by a modified LJ force expression in the high extension regime. The new in-plane force-extension relationship (referred to as the WLC-POW-LJ) is now:

$$F = \begin{cases} \frac{k_p}{L^m} - \frac{k_B T}{L_p} \left( \frac{1}{4(1-x)^2} - \frac{1}{4} + x \right), & L \leq L_{cut} \\ \frac{k_p}{L^m} + 24\varepsilon \left( 2 \frac{\sigma^{12}}{(L-L_{adj})^{13}} - \frac{\sigma^6}{(L-L_{adj})^7} \right), & L > L_{cut} \end{cases} \quad (3.22)$$

where  $\varepsilon$  is the energy of the spectrin-actin bond (the depth of the LJ energy potential well),  $\sigma$  is the characteristic interaction length scale,  $L_{adj}$  is the shift of the LJ force expression, and  $L_{cut}$  is point where the in-plane force expression switches from the WLC-POW portion to the POW-LJ portion of the expression. A comparison between the WLC-POW-LJ and the WLC-POW force-extension relationship is shown in [FIGURE 3-3](#). Note that the force resulting from the POW term is effectively zero at large extensions (such as when the force expression is in the range of  $L > L_{cut}$ ). For these cases, the POW force is not computed in simulations in the interest of increasing simulation speed.

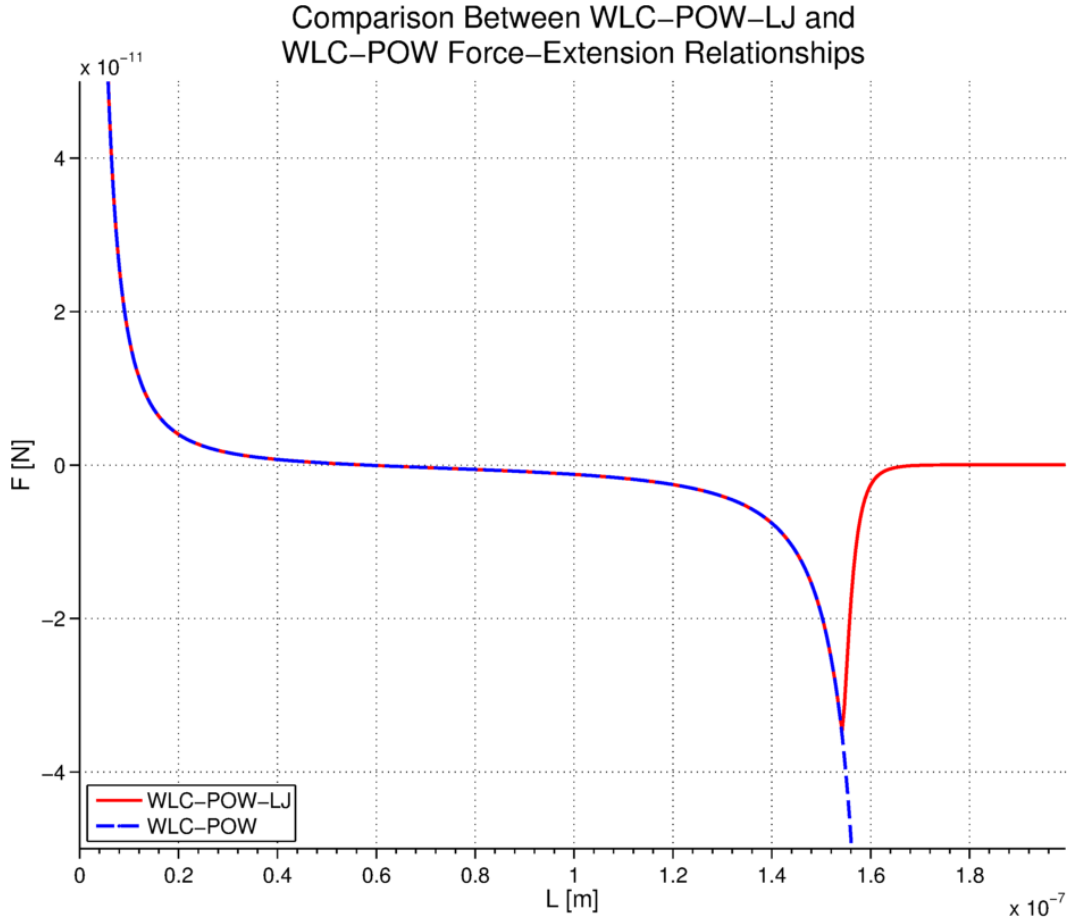


FIGURE 3-3: Comparison between WLC-POW-LJ and WLC-POW force-extension relationships.

The energy of the spectrin-actin bond is given by:

$$\varepsilon = k_B T \ln(K_{a,SA}) \quad (3.23)$$

where  $k_B$  is the Boltzmann constant,  $T$  is the temperature, and  $K_{a,SA}$  is the equilibrium association constant of the spectrin-actin bond. The characteristic interaction length scale is given by:

$$\sigma = \frac{2r_0}{2^{(1/6)}} \quad (3.24)$$

where  $r_0$  is the equilibrium distance of the bonds between topologically connected spectrin beads in a chain. The equilibrium distance of the bonds is given by:



$$r_0 = \frac{L_m}{S-1} \quad (3.25)$$

where  $S$  is the number of beads in the spectrin chain (39, in this case) and  $L_m$  is the contour length of the chain. The characteristic interaction length scale takes the form in [EQUATION 3.24](#) so that the equilibrium distance of the bond between the end spectrin bead of the chain and the actin bead is  $2r_0$ . The shift in the LJ force expression is given by:

$$L_{adj} = L_{cut} - L_{min} \quad (3.26)$$

where  $L_{min}$  is the location of the minimum value of the force in the LJ force expression (which is the location of the inflection point on the LJ energy potential well). The location of the minimum of the force in the LJ force expression is given by:

$$L_{min} = \sigma \left( \frac{26}{7} \right)^{(1/6)} \quad (3.27)$$

Finally,  $L_{cut}$  can be determined by finding the point on the WLC-POW force expression where the magnitude of the force is equal to the minimum value of the force in the LJ force expression:

$$\frac{k_p}{L_{cut}^m} - \frac{k_B T}{L_p} \left( \frac{1}{4(1-x_{cut})^2} - \frac{1}{4} + x_{cut} \right) = 24\epsilon \left( 2 \frac{\sigma^{12}}{(L_{cut})^{13}} - \frac{\sigma^6}{(L_{cut})^7} \right) \quad (3.28)$$

where  $x_{cut} = L_{cut}/L_m$ . Solving the above expression for  $L_{cut}$  completes the derivation of the new parameters in the WLC-POW-LJ force expression. The resulting force expression is continuous, but its derivative is not.

In the WLC-POW-LJ, it is possible for the separation distance between the corresponding actin nodes of a spectrin link to be greater than the contour length of the link. Physically, under these circumstances, the spectrin link is not being

stretched along with the actin nodes (the link is still inextensible, as was assumed in the formulation of the WLC model in [SUBSECTION 2.5.2](#)). Instead, a fracture has occurred, either between the spectrin link and one of the actin nodes or within the spectrin link itself. The implications resulting from this situation are described in detail in [SUBSECTION 3.4.5](#) and [SUBSECTION 3.4.6](#).

The hRBC is a viscoelastic object suspended in a viscous medium, so an accurate model of hRBC behavior must account for these effects. The effect from the membrane viscosity is modeled by adding a liner dashpot in the parallel with the spring representing the spectrin link (in-plane contribution to the force field). This arrangement is similar to the Kelvin-Voigt model of viscoelasticity [106], except the spring representing the spectrin link is nonlinear in the Combined hRBC model. The dissipative force acting on node  $i$  is given by:

$$\vec{F}_{D,r} = \frac{\sqrt{3}}{4} \eta_m (\vec{v}_j - \vec{v}_i) \quad (3.29)$$

where  $\eta_m$  is the membrane viscosity and  $v_{i,j}$  are the velocities of nodes  $i$  and  $j$ , respectively. This forces opposes changes in the relative velocities between nodes.

In addition to the dissipative force described above, a second dissipative force acting on the nodes is defined, based on the relative velocity between a particular node and the center of mass of the hRBC. In this formulation, the actin node is modeled as a sphere in Stokes flow. The dissipative force acting on node  $i$  is given by:

$$\vec{F}_{D,r} = 6\pi C_{eff} R_{eff} (\vec{v}_{cm} - \vec{v}_i) \quad (3.30)$$

where  $\vec{v}_{cm}$  is the velocity of the center of mass of the hRBC.  $C_{eff}$  is an effective viscosity given by:

$$C_{eff} = \sqrt{\eta_o \eta_i} \quad (3.31)$$

and  $R_{eff}$  is an effective radius given by:

$$R_{eff} = \sqrt[3]{\frac{3m_{actin}}{4\pi\rho_{actin}}} \quad (3.32)$$

In [EQUATION 3.31](#) and [EQUATION 3.32](#),  $\eta_o$  is the viscosity of the blood plasma,  $\eta_i$  is the viscosity of the cytoplasm,  $m_{actin}$  is the mass of a single actin node, and  $\rho_{actin}$  is density of an actin node. Including this additional term, in addition to making the Combined hRBC model more physically accurate, also helps to damp unusual high-frequency fluctuations that may occur in simulations and cause them to diverge.

### 3.3 Metabolic Model of the Human Red Blood Cell

The metabolic model used in this thesis is taken from “Modelling Metabolism with Mathematica” [15]. This model was chosen as the basis for representing the metabolism of the hRBC because it is one of the more comprehensive metabolic models found during a literature review that included the kinetics of the chemical reactions in the metabolism. In a dynamic simulation, the reaction kinetics are extremely important (just as the mass/viscosity/damping coefficient are important in a dynamic simulation of a purely mechanical system). Without the kinetic parameters, the system of ODE’s modeling the metabolism can only be used to solve for the steady-state concentrations of the metabolites (where the rates of change of the metabolite concentrations are zero).

The Modelling Metabolism with Mathematica (MMwM) model has 56 metabolites (participants in chemical reactions) and 53 chemical reactions. The metabolite concentrations are in terms of mol/L. The MMwM model includes reactions that are part of the glycolytic and pentose phosphate pathways. These two pathways are responsible for MgATP generation (from glucose in the bloodstream) and producing the necessary reducing equivalents. A metabolic map that helps to illustrate the metabolism is shown in [FIGURE 3-4](#).

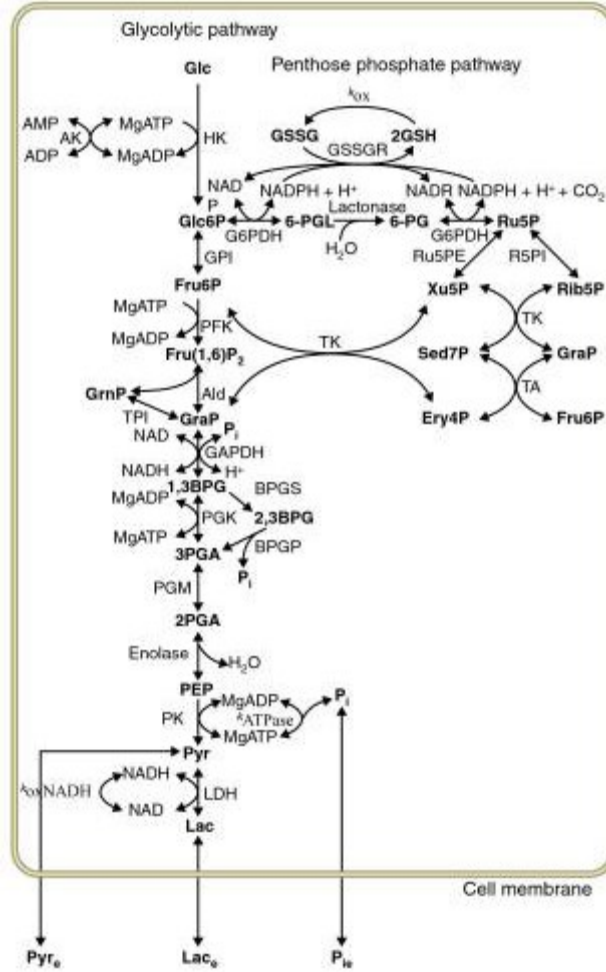


FIGURE 3-4: Metabolic map of hRBC metabolism [15]

The MMwM model was formulated with the intention of simulating the 2-3 BPG pathway (Rapoport-Luebering shunt), but it contains the basic reactions in the hRBC metabolism to a sufficient level of detail that it is able to be used in a simulation more concerned with MgATP production. The generation of MgATP is of special interest to this thesis because the mechanical behavior of the hRBC is connected to the metabolism through the MgATP concentration. This connection is explained in detail in [SECTION 3.4](#).

**FIGURE 3-5** shows the mathematical form of the hexokinase reaction rate in the MMwM model.

$$\begin{aligned}
 &K_i[hk, mgatp] = 1.0 \cdot 10^{-3}; \\
 &K_m[hk, mgatp] = 1.0 \cdot 10^{-3}; \\
 &K_i[hk, glc] = 4.7 \cdot 10^{-5}; \\
 &K_i[hk, glc6p] = 4.7 \cdot 10^{-5}; \\
 &K_i[hk, mgadp] = 1.0 \cdot 10^{-3}; \\
 &K_m[hk, mgadp] = 1.0 \cdot 10^{-3}; \\
 &K_{di}[hk, bpg] = 4.0 \cdot 10^{-3}; \\
 &K_{di}[hk, glc16p2] = 30.0 \cdot 10^{-6}; \\
 &K_{di}[hk, glc6p] = 10.0 \cdot 10^{-6}; \\
 &K_{di}[hk, gsh] = 3.0 \cdot 10^{-3}; \\
 &HK = 25 \cdot 10^{-9}; \\
 &kcatf[hk] := \\
 &\quad \frac{180 \cdot 1.662}{1 + (10^{-pH1[t]} / 10^{-7.02}) + (10^{-9.55} / 10^{-pH1[t]})} \\
 &kcatr[hk] := \\
 &\quad \frac{1.16 \cdot 1.662}{1 + (10^{-pH1[t]} / 10^{-7.02}) + (10^{-9.55} / 10^{-pH1[t]})} \\
 &hkrd := \left( 1 + \frac{MgATP[t]}{K_i[hk, mgatp]} + \frac{Glc[t]}{K_i[hk, glc]} + \right. \\
 &\quad \frac{MgATP[t]Glc[t]}{K_i[hk, glc]K_m[hk, mgatp]} + \frac{MgADP[t]}{K_i[hk, mgadp]} + \\
 &\quad \frac{Glc6p[t]}{K_i[hk, glc6p]} + \frac{MgADP[t]Glc6p[t]}{K_i[hk, glc6p]K_m[hk, mgadp]} + \\
 &\quad \frac{B23PG[t]Glc[t]}{K_{di}[hk, bpg]K_i[hk, glc]} + \frac{Glc16p2[t]Glc[t]}{K_{di}[hk, glc16p2]K_i[hk, glc]} + \\
 &\quad \left. \frac{Glc6p[t]Glc[t]}{K_{di}[hk, glc6p]K_i[hk, glc]} + \frac{GSH[t]Glc[t]}{K_{di}[hk, gsh]K_i[hk, glc]} \right); \\
 &V[hk] := Vol_i \cdot \frac{HK}{hkrd} \\
 &\quad \left( \frac{Kcatf[hk, Glc[t]MgATP[t]}{K_i[hk, glc]K_m[hk, mgatp]} - \frac{Kcatr[hk, Glc6P[t]MgADP[t]}{K_i[hk, glc6p]K_m[hk, mgadp]} \right);
 \end{aligned}$$

**FIGURE 3-5:** Mathematical form of hexokinase reaction rate in MMwM model [15].

The hexokinase reaction converts glucose to glucose 6-phosphate through the consumption of MgATP. The equation for the hexokinase reaction rate is included to illustrate the complexity that the equation for a single reaction can take (in the case of hexokinase, the reaction rate depends on 11 reaction constants and 7 metabolite concentrations). The reaction constants (listed at the top of **FIGURE 3-5**) are specific to the hexokinase reaction. In total, the MMwM model has 242 unique reaction constants, which are provided in the `f_metabolism.m` function in **APPENDIX A.1.6**.

In this thesis, the system of ODE's that represents the metabolism is numerically integrated using the MATLAB® function `ode15s`. The system is integrated each time step using the current metabolite concentrations as the initial conditions. After integration, the intermediate metabolite concentrations are discarded, and the final metabolite concentrations at the end of the time step are taken as the new metabolite concentrations. The new metabolite concentrations are then used as the initial conditions for the integration in the next time step.

In the MMwM model, the glucose concentration is held constant. This is justified by assuming that the concentration of glucose in the blood plasma is constant and the rate of the transport of glucose across the cell membrane occurs much more quickly than the rate of any other reaction in the metabolism. In addition, the intercellular pH as well as the extracellular concentrations of inorganic phosphate, lactate, and pyruvate are also held constant. The effect of holding the glucose concentration constant is that the cell never runs out of metabolic energy. There is a delay, however, between a decrease in the concentration of MgATP (occurring after significant consumption) and an increase in the rate of MgATP production. This delay is due to a signaling cascade that has to occur backwards through the metabolic network before the rate of glucose consumption can be increased (which will eventually lead to an elevated MgATP concentration).

By simulating the MMwM model using the initial conditions provided in the text, a steady-state condition can be achieved. This is a dynamic equilibrium, however, because the reactions are still taking place despite that the observed

metabolite concentrations are not varying. The authors of “Modelling Metabolism with Mathematica” suggest simulating the metabolism for  $1e6$  seconds ( $\sim 12$  days) of simulation time (not wall time) in order to get the metabolism to reach its steady-state. During subsequent simulations of the hRBC model used in this thesis, the steady-state metabolite concentrations are used at the initial conditions for the metabolite concentrations at the beginning of the simulation.



### 3.4 Relationship Between Mechanical and Metabolic Models

It has been known for some time that the metabolism of the hRBC has an effect on the mechanics of the hRBC [5, 107, 108, 109, 110, 111]. It was proposed, from a theoretical standpoint, by Gov et al. [112, 113] that the hydrolysis of MgATP at the actin nodes in the cytoskeletal network results in the phosphorylation of the spectrin links, thereby causing them to dissociate from the actin node (In the article by Gov et al., the interaction is described as being between ATP and spectrin. MgATP, however, is the functional form of ATP that participates in energy expenditure reactions). This proposition has been investigated in earnest, and is one of the leading hypotheses proposed to explain hRBC membrane behavior [114, 115]. The modeling efforts in this thesis build upon the proposition by Gov et al. The dissociation of the spectrin links may be driven by a stress-relaxation of the cytoskeletal network [116].

The dissociation of a spectrin link eliminates the in-plane interaction between the corresponding actin nodes. The other force field terms, however, are unaffected by the dissociation of the link and still act on both nodes. The dissociation of many links can result in changes in the global hRBC properties. Changes in global hRBC properties will result in changes the morphology of the hRBC. The hRBC can regulate the dissociation of spectrin links through its metabolism [117, 118, 119, 120]. Theories based on experimentally observed hRBC shape changes (from discocyte to echinocyte) under metabolically induced stress suggest that the loss of MgATP is linked to a stiffer cytoskeleton [83]. In the actual hRBC, the phosphorylation of the spectrin is reversible

(dephosphorylation) and the spectrin link can reassociate with an actin node (not necessarily the same one it originally dissociated from). This results in a network with a dynamic topology that is regulated by the concentration of MgATP in the cytoplasm of the hRBC [121]. The behavior of the hRBC model used in this thesis deviates slightly in this regard from the behavior of the actual hRBC. This deviation is in the interest of simplicity of the model, as is justified in the following subsections.

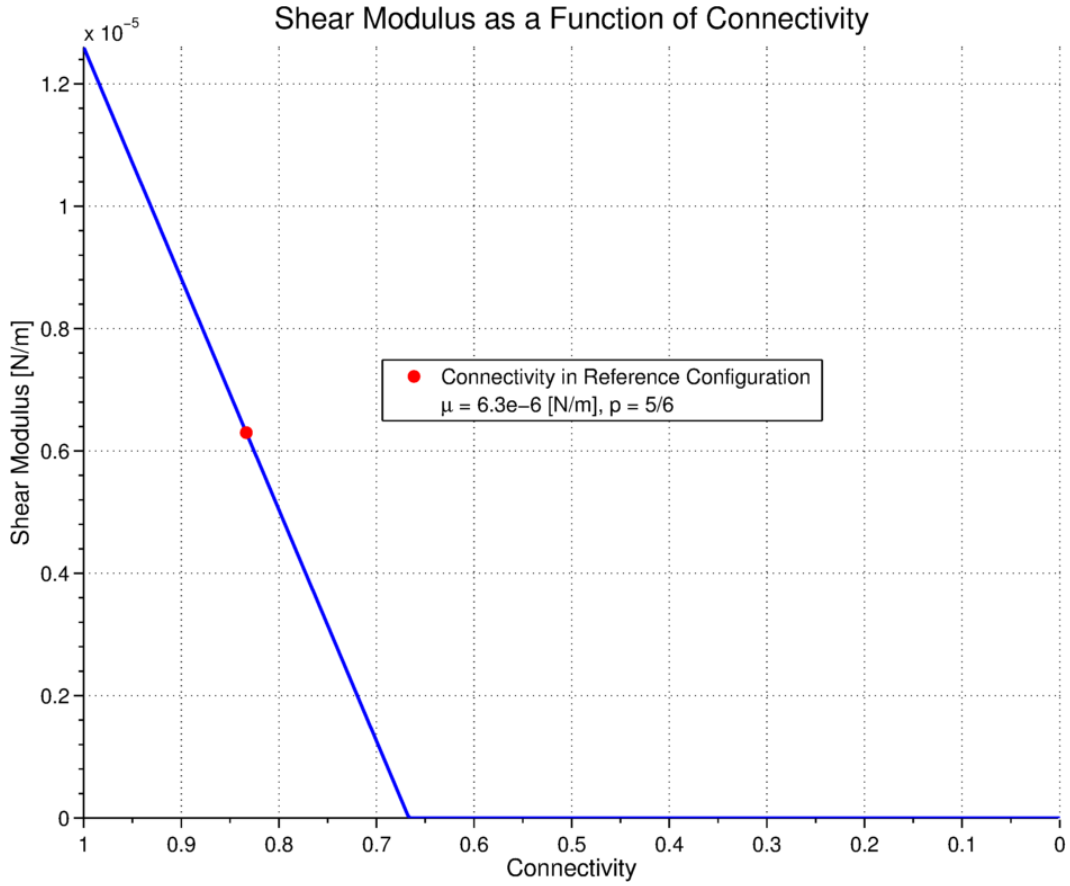
In order to combine the mechanical model of the hRBC with the metabolic model of the hRBC, modifications to both the CG-RBC model and the MMwM model need to be made so the models can be coupled and work in unison. The details of the connection between the models are described in the following subsections.

### 3.4.1 Mechanical-Connectivity Relationship

The relationship between the shear modulus of the hRBC and the proportion of associated spectrin links (as mentioned in [SECTION 2.6](#)) is given by:

$$\mu = \begin{cases} \mu_{max} \left( \frac{p-p_r}{1-p_r} \right), & p \geq p_r \\ 0, & p < p_r \end{cases} \quad (3.33)$$

where  $\mu_{max}$  is the maximum shear modulus of the hRBC,  $p_r$  is the rigidity percolation threshold (defined earlier in [SECTION 2.6](#)), and  $p$  is the proportion of associated spectrin links. For a fully intact network (all spectrin links connected)  $p = 1$ . The proportion of associated links is also referred to as the connectivity. A plot of the shear modulus as a function of connectivity is shown in [FIGURE 3-6](#).



**FIGURE 3-6:** Shear modulus as a function of the connectivity of the network. Note that the x-axis is inverted.

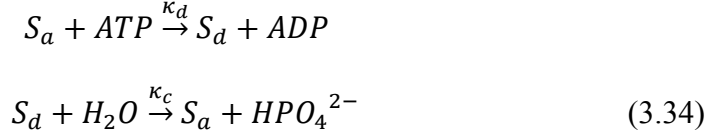
Note that the shear modulus decays linearly from its maximum value (at  $p = 1$ ) until the rigidity percolation threshold is reached (at  $p = 2/3$ ). The shear modulus is zero if the connectivity of the network is less than the rigidity percolation threshold.

Under the assumption that the maximum shear modulus of the hRBC is twice the shear modulus observed in experimental conditions ( $\mu_{max} = 2\mu$ ) [112], the connectivity of the hRBC in its reference configuration can be found to be  $p_0 = 5/6$  (where the subscript indicates that the hRBC is in its reference configuration). This means that, when the hRBC is at rest in its reference

configuration, only  $5/6$  of its links are associated. This corresponds to 68,355 links in the hRBC model used in this thesis.

### 3.4.2 Metabolic-Connectivity Relationship

Zhang et al. [78] defined a “naive” model of spectrin phosphorylation with the following kinetic equations:



where  $S_a$  is the associated form of the spectrin links,  $S_d$  is the dissociated form of the spectrin links,  $\kappa_d$  is the rate of the dissociation reaction, and  $\kappa_c$  is the rate of the reassociation reaction. This model of spectrin phosphorylation is consistent with other models of the reaction in literature [122]. The relationship between the concentration of dissociated spectrin links and the concentration of associated spectrin links is given by:

$$[S_d] = [S_T] - [S_a] \quad (3.35)$$

where  $[S_d]$  is the concentration of dissociated links,  $[S_T]$  is the total concentration of spectrin links in the hRBC (defined by the number of spectrin links in the model,  $N_s$ ), and  $[S_a]$  is the concentration of associated spectrin links. The proportion of associated links in a network that is governed by this set of reactions is given by:

$$p = \frac{[S_a]}{[S_T]} \quad (3.36)$$

Under the assumption that the dissociations and reassociations of the spectrin links are independent, the proportion of associated spectrin links is the probability that a randomly selected link in the network will be in an associated state. This is the same probability that was defined in [SECTION 2.6](#).

By treating the reactions in [EQUATION 3.34](#) as a simple two-state kinetic model (assuming that only the concentrations of associated and dissociated spectrin links vary in the reaction) and invoking the law of mass action, a relationship between the reaction rates and the connectivity of the network can be determined at steady-state:

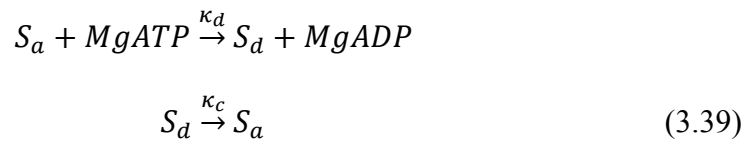
$$p = \frac{\kappa_c}{\kappa_c + \kappa_d[ATP]} \quad (3.37)$$

Under the same assumptions, the equilibrium association constant of the two reactions is given by:

$$K_a = \frac{\kappa_c}{\kappa_d[ATP]} \quad (3.38)$$

This equilibrium association constant was introduced earlier (in [SUBSECTION 3.2.2](#)). It is for the reaction between the spectrin links and the actin node. Note how the equilibrium association constant is inversely proportional to the ATP concentration.

The hRBC model used in this thesis uses a modified set of the reactions in [EQUATION 3.34](#):



In this modified version, the additional metabolites in the second reaction have been eliminated (since these metabolites do not appear in the MMwM model). Also, ATP and ADP have been replaced by MgATP and MgADP, respectively. The relations derived above are still valid for this modified set of reactions. The reactions in [EQUATION 3.39](#) are integrated into the set of reactions in the MMwM

model so that the metabolic model can account for dissociations and reassociations of the spectrin links.

Utilizing the result from [SUBSECTION 3.4.1](#) and [EQUATION 3.37](#), a relationship between the connectivity of the network and the concentration of MgATP can be derived. This relationship assumes an equivalence between the reference configuration of the hRBC (when it is at rest) and the metabolism of hRBC (when it is at steady-state). This connectivity of the network when the metabolism is at steady-state is given by:

$$p_0 = \frac{\kappa_c}{\kappa_c + \kappa_d[MgATP]_0} \quad (3.40)$$

where the subscript on the MgATP concentration indicates that it is the steady-state concentration of MgATP. [EQUATION 3.40](#) can be solved for  $\kappa_c$ , yielding:

$$\kappa_c = 5\kappa_d[MgATP]_0 \quad (3.41)$$

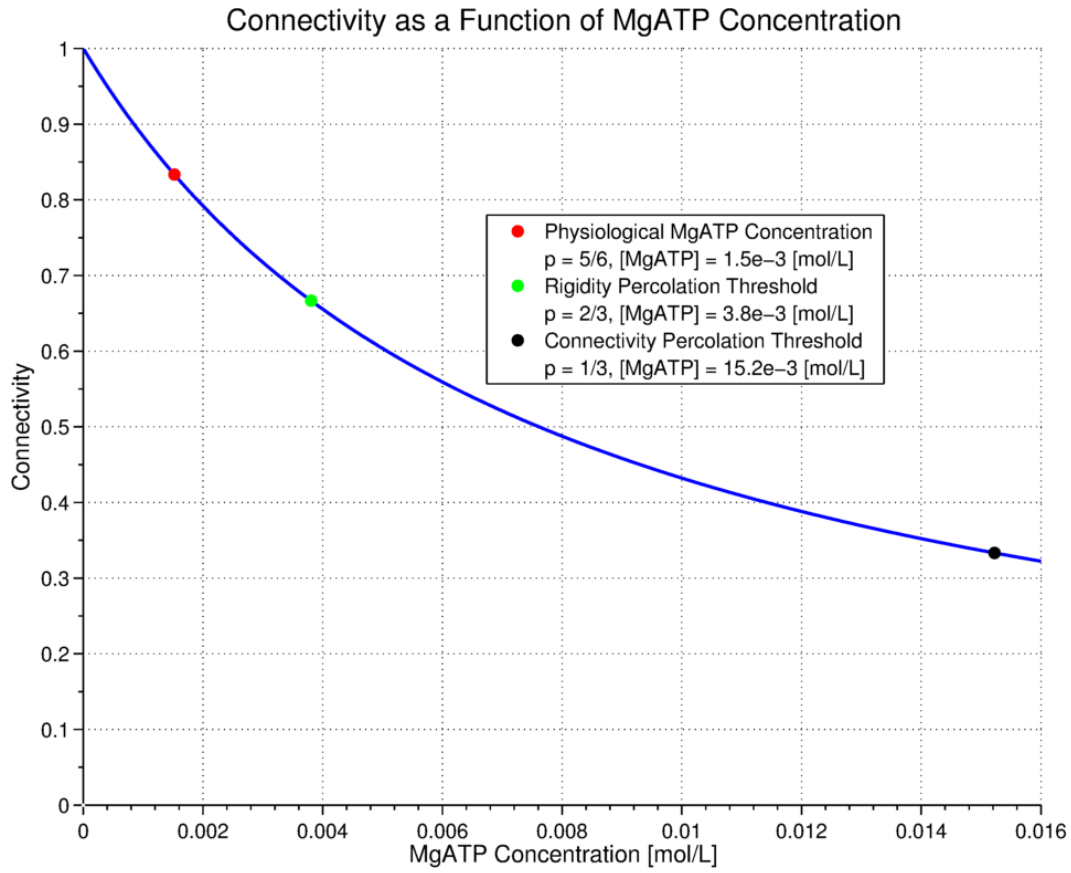
By substituting this result back into [EQUATION 3.37](#), the relationship between the connectivity of the network and the MgATP concentration can be defined. This relationship is given by:

$$p = \frac{5[MgATP]_0}{5[MgATP]_0 + [MgATP]} \quad (3.42)$$

The form of [EQUATION 3.42](#) ensures that  $p = p_0 = 5/6$  when

$[MgATP] = [MgATP]_0$ . A plot of the connectivity as a function of the MgATP concentration is shown in [FIGURE 3-7](#).





*FIGURE 3-7: Connectivity as a Function of MgATP Concentration*

Note that the relationship is nonlinear. The slope is steep in the region near the physiological MgATP concentration, but flattens out as the MgATP concentration increases. Although it is not used in this thesis, the connectivity percolation threshold is indicated on this plot for comparison reasons.

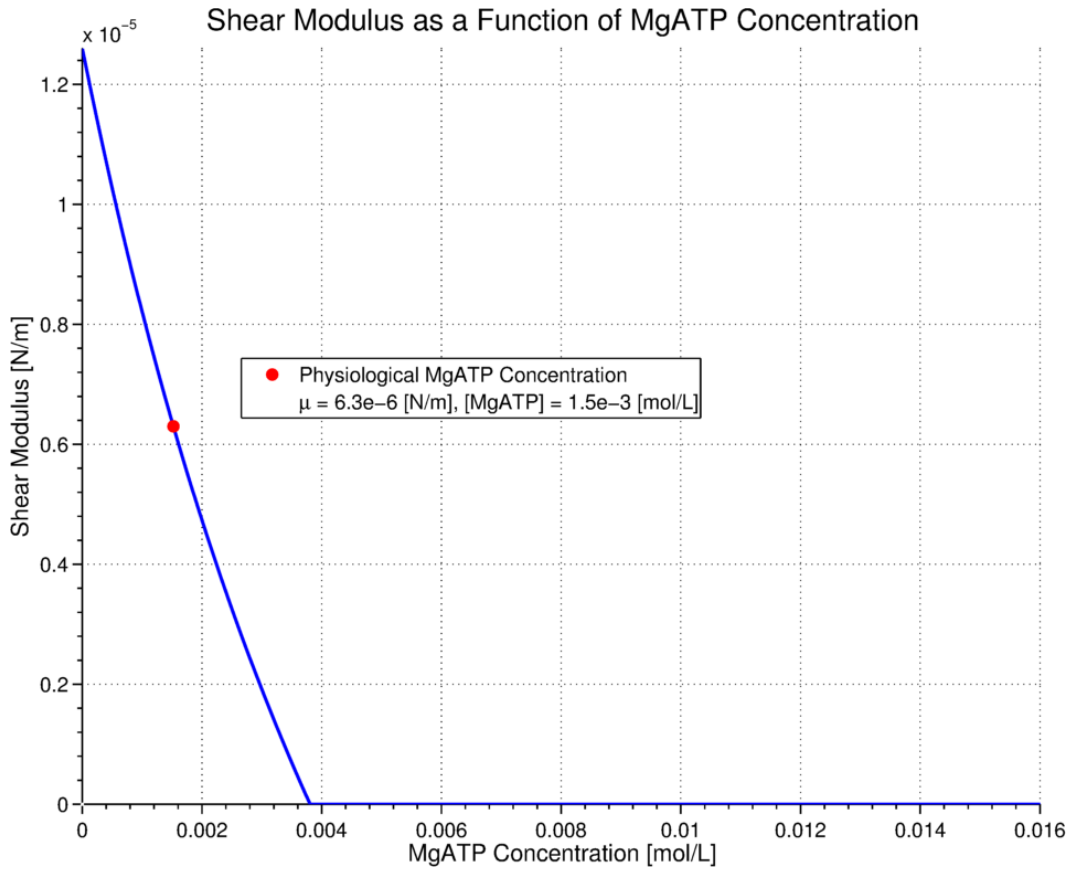
A very important stipulation of the hRBC model used in this thesis is that the spectrin link can only reassociate with the same actin node it originally dissociated from. Other spring network models [101, 123] allow for “links” to reassociate with different nodes. Topological remodeling of the cytoskeleton to the same degree as was done in the above articles adds unnecessary complication to the hRBC model, and is ultimately beyond the scope of this thesis.

### 3.4.3 Mechanical-Metabolic Relationship

A relationship between the shear modulus of the hRBC and the concentration of MgATP can be defined. This relationship is given by:

$$\mu = \begin{cases} \mu_{max} \left( \frac{[MgATP]}{(5[MgATP]_0 + [MgATP])(p_r - 1)} + 1 \right), & [MgATP] \leq 3.8e^{-3} \text{ mol/L} \\ 0, & [MgATP] > 3.8e^{-3} \text{ mol/L} \end{cases} \quad (3.43)$$

The cutoff MgATP concentration used above is the concentration of MgATP that would cause the connectivity of the network to be equal to the rigidity percolation threshold ( $p = 2/3$ ). A plot of the shear modulus as a function of the MgATP concentration is shown in **FIGURE 3-8**.



**FIGURE 3-8:** Shear modulus as a function of MgATP concentration.

Note that the relationship between the shear modulus and the MgATP concentration is not linear. This is due to the nonlinear relationship between the

connectivity and the MgATP concentration. The shear modulus predicted by **EQUATION 3.43** is consistent with the stiffer cytoskeleton observed under MgATP depletion [83]. **EQUATION 3.43** is not used as part of the simulation of the hRBC, but it is included so as to complete the derivation of the mechanical-metabolic relationship.

### 3.4.4 Discrete-Continuum Transformation

Since the results of simulating the metabolism of the hRBC produces “continuum” results (concentration of associated spectrin links in units of mol/L) and the links that make up the hRBC cytoskeleton are discrete entities, a connection between the concentration of associated spectrin links and the number of associated spectrin links is needed. This mapping between the continuum and discrete domains is achieved by converting the number of links to a concentration. The maximum concentration of spectrin links for a given hRBC volume is given by:

$$[S_T] = \frac{N_s}{N_A V_L} \quad (3.44)$$

where  $N_A$  is Avogadro’s constant (in units of links/mole in this context) and  $V_L$  is the volume of the hRBC (in liters). This transformation produces a concentration in units of moles/liter. (Note, in the simulation of the hRBC, the volume of the hRBC is calculated in terms of cubic meters. This volume needs to be converted to liters before the transformation to concentration can be performed). Using [EQUATION 3.44](#), the concentration of associated links can be determined. The associated link concentration is given by:

$$[S_a] = p[S_T] = p \frac{N_s}{N_A V_L} = \frac{N_{asc}}{N_A V_L} \quad (3.45)$$

where  $N_{asc}$  is the number of associated links.

This transformation offers a simple method of converting from the discrete domain to the continuum domain. If the connectivity and the total number of links in the network are known, the concentration of associated spectrin links can

be determined using **EQUATION 3.45**. However, if the concentration of associated spectrin links and the connectivity are known, **EQUATION 3.45** can only be used to determine the number of associated links, not which specific links are associated. This is because information about the links is lost in the discrete to continuum transformation. Additional provisions are required in the Combined hRBC model in order to preserve the information about the connectivity of specific links through the transformation, as well as to determine the changes in the connectivity after the metabolic simulations outputs changes the concentration of associated links.

### 3.4.5 Continuum-Discrete Transformation

The number of associated links is calculated using [EQUATION 3.45](#) and the concentration of associated links is output from the metabolism simulation. This is the number of associated links as dictated by the metabolism. The metabolically-dictated number of associated links is then compared to the actual number of associated links. A discrepancy between the metabolically-dictated number of associated links and the actual number of associated links may exist because the simulations are dynamic, and the connectivity of the network is allowed to vary. In the case where the metabolically-dictated number of associated links is less than the actual number of associated links, then a specified number of links (equal to the difference between the metabolically-dictated number of associated links and the actual number of associated links) are dissociated so that the actual number of associated links matches the metabolically-dictated number of associated links. In the case where the metabolically-dictated number of associated links is greater than the actual number of associated links, dissociated links are not reassociated in order to meet this requirement (the metabolism can only dissociate links, not reassociate link).

The reason the link reassociation process cannot directly follow the results from the reassociation reaction is rooted in the activation-controlled vs. diffusion-controlled nature of the spectrin phosphorylation and dephosphorylation reactions. In reality, once a link dissociates, the spectrin filament no longer experiences a force from the actin node it dissociated from. This causes the filament to drift towards a configuration where the end-to-end length of the

filament is close to the equilibrium length of the filament. This drifting will likely cause a separation between the end of the filament and the previously associated actin node. After the phosphorylation of the spectrin filament, it should be unable to immediately reassociate with the actin node. Only after the spectrin filament has been dephosphorylated can it reassociate with the actin node. This dephosphorylation process should take some time. During this “down time” the spectrin filament is drifting away from the actin node. After the spectrin filament has been dephosphorylated, it is some distance away from the actin node and has to drift back toward the node in order to reassociate with it. The time between when the spectrin link is dissociated and when it is able to reassociate is referred to as the wait time. Because the spectrin link is only able to reassociate with the actin node it originally dissociated from, rapid dissociation and reassociation between the spectrin link and the actin node can occur if there is no wait time between the dissociation of the spectrin link and its reassociation. This unphysical behavior is not desirable, especially in simulations being performed to determine the mechanical properties as a function of network connectivity. In simulations performed as a part of this thesis, the wait time was selected rather arbitrarily, but it should be directly related to the reaction rate of the spectrin-actin dissociation/reassociation reactions and the relaxation time of the spectrin link. Now that the idea behind the link selection has been introduced, the details of the selection process will be explained.

The specific links that are dissociated to meet the connectivity requirement set by the metabolism are selected using a weighted random selection without replacement (WRSWOR) algorithm. The basis for the selection of links is random because the exact details of the spectrin phosphorylation and dephosphorylation reactions are unknown. Using the equal a priori probability postulate, the probability can be related to the energy of the link. In the hRBC model used in this thesis, the potential energy stored in a link is only a function of the extension of that link (the individual link's "microstate"). In the absence of information regarding the spectrin phosphorylation and dephosphorylation reactions and the nanoscale behavior of a free spectrin filament, the best assumption regarding spectrin link selection is that the selection process is weighted by the extension of the spectrin links. Because configurations with a lower Helmholtz free energy are more desirable, it is assumed that the probability that stretched links will be selected is greater than the probability that less stretched links will be selected, since the dissociation of these stretched link will result in a greater reduction in the potential energy of the hRBC model. The selection is performed without replacement to ensure that links are not selected multiple times (the chance that a link is selected multiple times decreases as the number of links in the network increases).

When a spectrin link is selected for dissociation by the WRSWOR algorithm, the connectivity of that spectrin link is set to zero. This is equivalent to turning off the attractive portion of the WLC-POW-LJ force-extension relationship. The weight of a selected spectrin link (probability it will be selected



by the WRSWOR algorithm) is set to zero to prevent it from being selected multiple times. A wait time counter is also started when the spectrin link is selected. This counter starts at the specified wait time and decreased by the time step length each time step after the spectrin link is dissociated. This dissociation portion is repeated multiple times for every spectrin link that is selected to be dissociated during the current time step. After the required number of links are dissociated, the updated number of associated links is used to back calculate the concentration of associated links using [EQUATION 3.45](#). This is to ensure consistency in the simulation. The updated associated link concentration is used in the simulation of the metabolism.

When the wait time counter of a spectrin link reaches zero, the current separation distance between the actin nodes corresponding to the spectrin link is checked against the contour length of the spectrin filament. In the case where the separation distance is less than the contour length of the spectrin filament, the spectrin link is allowed to reassociate. When the spectrin link reassociates, the connectivity of the spectrin link is set to one, which is equivalent to turning back on the attractive portion of the WLC-POW-LJ force-extension relationship. The selection weight of the reassociated spectrin links is also updated using the current separation distance between the actin nodes that correspond to the spectrin link.

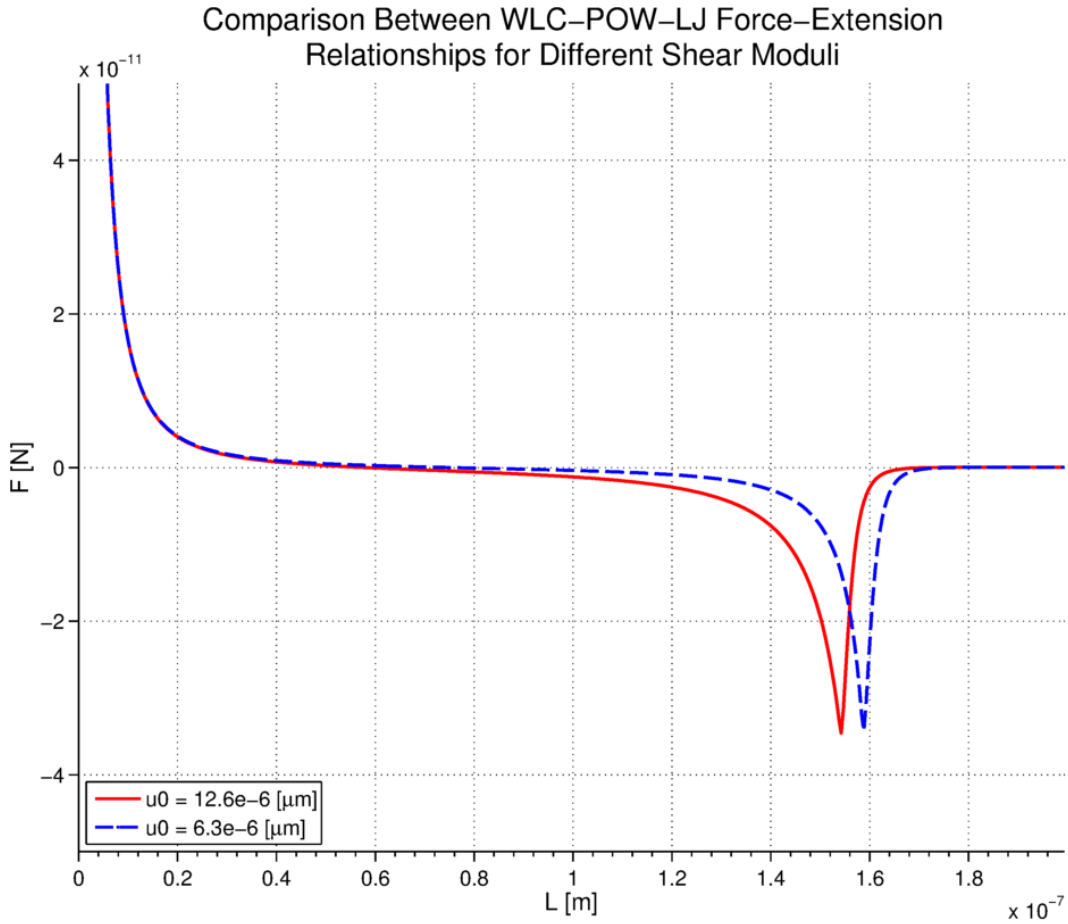
In the case where the separation distance is greater than the contour length of the spectrin filament, the spectrin link is not allowed to reassociate. The connectivity of the spectrin link stays at zero, preventing attractive interactions between the actin nodes corresponding to the spectrin link. The separation

distance between the corresponding actin nodes is checked against the contour length every time step, and the connectivity of the spectrin link remains at zero until the separation distance is less than the contour length. At this point, the spectrin filament is allowed to reassociate as described previously. Spectrin links that cannot reassociate after their wait time is up still count as dissociated links.

The selection of a specific link to dissociate is independent of the dissociation and reassociation history of the specific spectrin link, the connectivity of any of the other spectrin links in the network, and the dissociation and reassociation history of any of the other spectrin links in the network. The dissociation of a particular spectrin link does not affect the evaluation of any of the other force field terms. The use of the WRSWOR algorithm makes the Combined hRBC model in this thesis nondeterministic.

### 3.4.6 Additional Modification to CG-hRBC Model

Note that the use of the larger shear modulus requires a rederivation of a significant number of the force field parameters. The use of the rederived parameters in the force field results in a modified WLC-POW-LJ force-extension relationship. This modified relationship is shown in **FIGURE 3-9**.



**FIGURE 3-9:** Comparison between WLC-POW-LJ force-extension relationships for different shear moduli.

The red curve in **FIGURE 3-9** illustrates the force-extension relationship of the in-plane portion of the force field that is used in this thesis. Note how the larger shear modulus results in a stiffer WLC-POW-LJ force-extension response of the spectrin link. As a consequence of this, the location of the maximum attractive force of the stiffer WLC-POW-LJ force-extension relationship is shifted to a

shorter extension. Likewise, the location of the equilibrium position (location where force is equal to zero) is shifted to a shorter extension.

An additional modification to the WLC-POW-LJ force-extension relationship is incorporated in the hRBC model used in this thesis in order to account for damage incurred by spectrin-actin junction when the spectrin is stretched beyond its contour length. This modification permanently “turns off” the attractive portion of the WLC-POW-LJ force-extension relationship, leaving only the repulsive POW portion. The switch from the WLC-POW-LJ force-extension relationship to only the repulsive POW force-extension relationship for a particular link occurs when that link is stretched to a length greater than its contour length. A link that has undergone this switch is referred to as a “broken” link. In addition to having the attractive portion of the WLC-POW-LJ force-extension relationship permanently turned off, broken links also do not contribute to the concentrations of spectrin links. The number of broken spectrin links is subtracted from the total number of links in the Combined hRBC model, since these spectrin links effectively no longer exist. The selection weight of broken links is also set to zero, since they cannot be selected for dissociation by the WRSWOR algorithm.

There is a caveat, however, in the breaking of spectrin links. In the case when a link is dissociated (due to phosphorylation of the spectrin link because of the metabolism), and the separation distance between the corresponding actin nodes then increases, while the link is dissociated, to a separation greater than the contour length of the link, the link is not broken. Instead, the link remains

dissociated until its wait time counter reaches zero. At this point, the spectrin link either reassociates (if the separation distance between its corresponding actin nodes is less than the contour length of the link) or remains dissociated until the separation distance is less than the contour length of the link. This is the same as the treatment introduced in [SUBSECTION 3.4.5](#) for the reassociation of links where the separation distance between their corresponding actin nodes increases to lengths greater than the contour length (which would prevent the link from reassociating when the wait time counter finishes). Basically, this caveat ensures that dissociated links cannot be broken, regardless of separation distance. Keeping this treatment still allows the for the Combined hRBC model to be able to recover from larger deformations and return to its reference configuration.

### 3.5 Simulation of the Human Red Blood Cell

In [SECTION 3.1](#), the computation methods used create the initial triangulation of the hRBC model were introduced. In this section, the specific computational methods used to simulate the models introduced in [SECTION 3.2](#) and [SECTION 3.3](#), along with the interaction between the models introduced in [SECTION 3.4](#), are discussed. The coupling of the mechanical CG-hRBC model with the metabolic MMwM model through the connectivity of the cytoskeleton forms the hRBC model that will be used for simulations in this thesis. From this point on, this model will be referred to as the Combined hRBC model.

MD simulations on the Combined hRBC model can be performed to determine the response of the model to external conditions. These MD simulations on the Combined hRBC model are performed using a modified version of the Verlet algorithm introduced earlier in [SECTION 2.4](#). This modified version is known as the Velocity Verlet (it is similar to the Leapfrog method).

The form of the Velocity Verlet is:

$$\begin{aligned}
 \vec{r}_{t+\Delta t} &= \vec{r}_t + \vec{v}_t \Delta t + \frac{1}{2} \vec{a}_t \Delta t^2 \\
 \tilde{\vec{v}}_{t+\lambda \Delta t} &= \vec{v}_t + \lambda \vec{a}_t \Delta t \\
 \vec{a}_{t+\Delta t} &= fun(\vec{r}_{t+\Delta t}, \tilde{\vec{v}}_{t+\lambda \Delta t}) \\
 \vec{v}_{t+\Delta t} &= \vec{v}_t + \frac{1}{2} (\vec{a}_t + \vec{a}_{t+\Delta t}) \Delta t
 \end{aligned} \tag{3.46}$$

The  $\tilde{\vec{v}}_{t+\lambda \Delta t}$  term is an intermediate velocity, at a time between  $t$  and  $t + \Delta t$ .  $\lambda$  is a weighing factor that determines the time the intermediate velocity is calculated at ( $0 < \lambda < 1$ ).  $\lambda$  is usually taken to be 0.5 (this means that the intermediate velocity

is calculated exactly in the middle of the time step). The “*fun*” notation indicates that the acceleration at the  $t + \Delta t$  is a function of the position at  $t + \Delta t$  and the intermediate velocity. The Velocity Verlet is used instead of Störmer’s method because the velocities are needed if dissipative interactions are to be calculated. To decrease the amount of real-time it takes a simulation to run, the largest time step for which the simulation remains stable should be used. The optimization of the computation is discussed in detail in [SECTION 4.6](#). Ultimately, the proper time step length to use in simulations performed as a part of this thesis was determined heuristically.

### 3.5.1 Simulation Setup

Before simulations can be performed with the Combined hRBC model, the rest of the parameters required in the simulations need to be determined. The parameters used in the mechanical portion of the model are calculated and/or declared in the *hRBC\_Parameter\_Derivation.m* script file. This script loads the results from the *hRBC\_Triangulation.m* script file (*hRBC\_Triangulation.mat* file) in order to calculate the rest of the force field parameters. All the parameters are then saved in the *hRBC\_Parameters.mat* file.

### 3.5.2 Energy Minimization

A potential energy minimization needs to be performed on the Combined hRBC model using the node locations saved in *hRBC\_Triangulation.mat* as initial locations. As described in [SECTION 2.4](#), this minimization is performed at zero Kelvin (which means that the velocity of each node is set back to zero) and the Velocity Verlet described above is used to update the positions of the nodes every time step. The zero Kelvin condition is an attempt to ensure that an actual global minimum of potential energy is reached. Only the mechanical model is simulated during this minimization. (A true energy minimization cannot be performed while the metabolism is being simulated because the metabolism is continuously adding chemical energy to the system, in the form of MgATP).

A minimization of the Combined hRBC model is then performed. The MMwM model is simulated using the initial conditions provided in [15] (as described in [SECTION 3.3](#)). The mechanical model is simulated using the potential energy minimization method described above and allowed to interact



with the metabolic model. The interaction is necessary due to the coupling between the models. This minimization is also performed until both a potential energy minimum of the mechanical model and steady-state conditions for the metabolic portion are reached. The results of the second minimization are saved in *hRBC\_Minimized.mat*. This includes the connectivity data from the triangulation, the node locations, the surface area and volume, and the steady-state metabolite concentrations at the end of the minimization.

### 3.5.3 Simulation Execution

After the Combined hRBC model has been minimized, a simulation can be performed using the data from *hRBC\_Minimized.mat* as initial conditions. Because the minimization is performed at zero Kelvin, the initial velocity of all nodes is set to zero. An example simulation is described in detail in this section in order to illustrate the simulation process. The simulation is executed using the *hRBC\_Stretching\_Simulation.m* script file, which serves as a wrapper for the function files that perform the calculations in the simulation. The example simulation is the computational equivalent of an optical tweezers stretching experiment of a hRBC [32, 33, 39, 41]. Forces are applied to specific nodes in the Combined hRBC model. These forces perturb the model away from its reference configuration, towards a new configuration that minimizes the potential energy. The three sections (and relevant subsections) of *hRBC\_Stretching\_Simulation.m* script file are described in detail below.

The first section of the code file sets up the simulation. First, the date and time at the beginning of the simulation are saved. Next, global variables are declared and the data from *hRBC\_Parameters.mat* and *hRBC\_Minimized.mat* is loaded. Then, flags for simulation options are set. These flags are logicals that turn on/off the stretching simulation as well as the saving, plotting, and exporting (of the plotted results) of simulation results. Next, the number of time steps, length of each time step, and the applied force in the simulation are defined. The integration time step weighing factor ( $\lambda$  in [EQUATION 3.46](#)) is also defined. Next, all matrices and vector not already defined are initialized to zero. The wait times of currently dissociated links are set using times selected from a uniform random distribution. The selection weights for all links is calculated using their current extensions. Then, the nodes that the stretching force is applied to are selected, and the force is distributed evenly between these nodes. This completes the first section of the code file.

The second section of the *hRBC\_Stretching\_Simulation.m* code file performs the actual time step iterations in the simulation process. Each individual time step has five subsections. In the first subsection, some variables are updated for use in the current time step, links that are supposed to be reassociating (wait times are up) are reconnected, and links that are supposed to be dissociating due to non-metabolic reasons are disconnected. In the second subsection, the metabolism is simulated. In the third subsection, the number of links that are to be dissociated due to the metabolism is calculated, and the specific links are selected and disconnected. In the fourth subsection, the mechanical portion of the

Combined hRBC model is simulated. In the fifth subsection, the rest of the variables are updated in preparation for the next time step. The specifics of each subsection are described next.

In the first subsection, the time step span for the current time step is set (for use in the numerical integration of the metabolism). Then, the wait times of dissociated spectrin links are updated. Next, currently connected spectrin links that have an extension greater than the contour length of the spectrin filament are permanently broken (achieved by setting their connectivities to zero). Then, spectrin links that are reassociating during the current time step are allowed to reassociate and their connectivities are updated. Next, the selection weights are updated for the newly reassociated spectrin links using the current separation distance between the actin nodes associated with each link. Then, a check is performed to ensure that a spectrin link that has been broken previously still has a connectivity of zero. Next, using the number of links that have been broken, the maximum possible number of spectrin links is calculated and convert to a maximum concentration.

In the second subsection, the metabolism is then simulated using *ode15s* and the time span calculated at the beginning of the time step. The concentration of all metabolites is updated to reflect the concentrations obtained at the end of the time span (“final” concentrations from the output from *ode15s*).

In the third subsection, the concentration spectrin links (determined from the metabolism) is then used to calculate the number of links that need to be dissociated so that the actual concentration of spectrin links is equal to the desired

concentration of spectrin links (as determined from the metabolism). The number of links to dissociate (*num\_dis* in code) is checked to ensure that it is a valid number of nodes (integer, numeric, positive). If the number of nodes to dissociate is invalid, the simulation is halted and the current simulation data is saved to an error dump. If the number of nodes to dissociate is valid, the simulation proceeds. The actual links that are dissociated (using the WRSWOR algorithm introduced in [SUBSECTION 3.4.5](#)) using the *datasample* function in MATLAB®. This function takes the number of links to dissociate, a vector containing the unique identifying numbers of all the links, and a vector of the selection weights of the links. It also has an option for selection with or without replacement (in its usage in this thesis, the selection without replacement is used). The *datasample* function then outputs a vector containing the identifying numbers of the links that have been selected for dissociation. This vector is fed into the *f\_diss.m* function, which performs the dissociation of the selected links. In the *f\_diss.m* function, the connectivities of the selected links and the selection weights are set to zero, and the wait time counters for the links are set to the specified wait time. Another check is performed to determine if the number of nodes dissociated by the metabolism causes the connectivity of the network to decrease to less than half of the maximum possible value. (Through testing, it has been determined that simulations using reasonable stretching forces in which the connectivity drops to such a low value are unstable, or on the brink of diverging). If this is the case, the simulation is halted and the current simulation data is saved to an error dump. If the connectivity is valid, the simulation proceeds.

In the fourth subsection, the MD portion of the simulation is performed. The nodal mass is calculated using the volume and density. The new accelerations are calculated (using the updated total forces determined in the previous time step). The new positions are then calculated using the current accelerations and the previous velocities. The intermediate velocities are also calculated using the current forces and the previous velocities. The updated forces are then calculated using the new positions and intermediate velocities. The conservative forces are calculated using the *f\_cons\_forces.m* function. This function uses the positions and connectivity to calculate the forces on the nodes resulting from the terms in the force field ([EQUATION 3.6](#)) using the equations from Appendix A of [93]. The contributions to the free energy due to the links, the local area, and the bending are also calculated and output. In addition, the link lengths, the areas of individual elements, the total area, and the total volume are calculated and output. The dissipative forces are calculated using the *f\_diss\_forces.m* function with the intermediate velocities as input. The total forces acting on the Combined hRBC model are then calculated as the sum of the conservative forces, the dissipative forces, and the applied stretching forces. This updated total force is used next time step to calculate the accelerations. The energetic contributions from each force field term are also calculated and saved (so that the energy of the model as a function time can be plotted and analyzed later). The total potential energy is checked, and if it is above a threshold value, the simulation is halted and the current simulation data is save to an error dump. An unusually high total potential energy typically indicates that one or more

nodes has been “ejected” from the model due to an artificially high force (most likely from the error dissipative forces). If the total potential energy is below this threshold value, the simulation proceeds.

In the fifth subsection, the variables required for the next time step are updated. These variables included the nodal mass of the hRBC (calculated from the updated volume), the maximum concentration of spectrin links, and the current concentration of associated spectrin links. The wall time elapsed during the simulation up to this point is also calculated. This completes the second section of the code file.

The third section of the *hRBC\_Stretching\_Simulation.m* code file performs the post-processing calculations of the simulation results, plots the results (if desired), and saves the results. The date and time of the end of the simulation are also recorded. In the post-processing calculations, the averages energies, axial and transvers diameters, triangulation quality, and relative shape anisotropy are calculated. The wall time per time step is also calculated for simulation evaluation. The simulation results that can be plotted are shown in [SECTION 4.1](#). The results of the simulation are saved to .mat file with a filename specific to the individual simulation. The form of the filename is *DATA\_Stretching\_Simulation\_\_“number of time steps in simulation”\_\_“length of each time step”\_\_“ending date and time of simulation”.mat*. This completes a simulation of the Combined hRBC model. The plots of the results from this example simulation of the Combined hRBC model are presented in [SECTION 4.1](#).

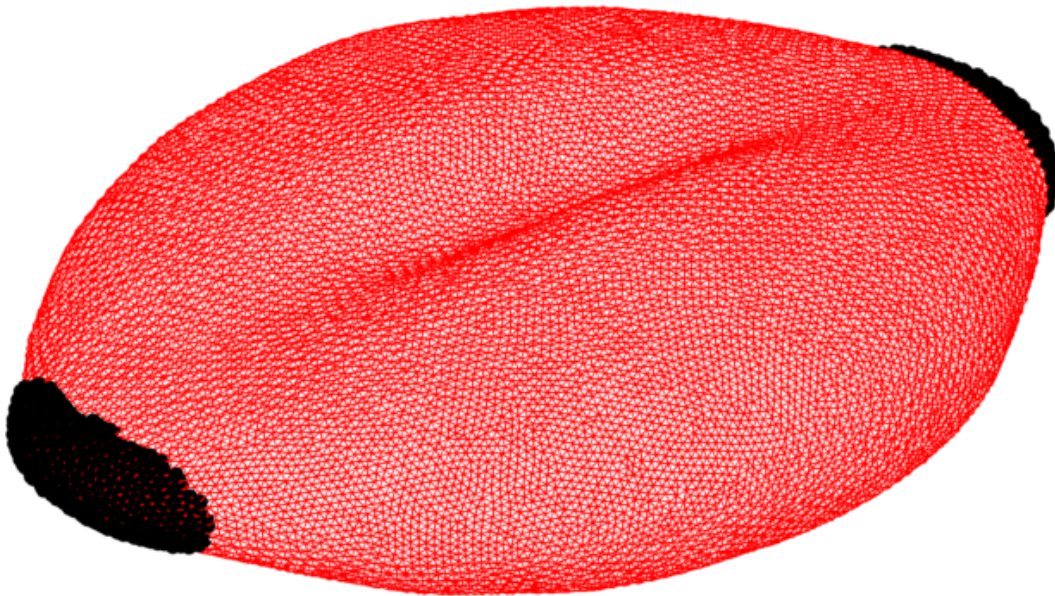
## Chapter 4: Results, Discussion, and Analysis

### 4.1 Simulation Environment

A major result from the efforts put forth in this thesis was the creation of a unified simulation environment in MATLAB®. A comprehensive simulation of the hRBC, from mesh creation and triangulation to plotting and visualization of the simulation results, can be performed utilizing the set of code files provided in [APPENDIX A.1](#). The functionality of the simulation environment was described in [SECTION 3.5](#). This section presents plots of the results from the example simulation used in [SUBSECTION 3.5.3](#), beginning on the next page. Some basic observations from the data are presented in conjunction with the plots. This stretching simulation was run for 100,000 time steps (with a time step length of 100 nanoseconds) under an applied force of 100 piconewtons.

**FIGURE 4-1** is a plot of the final configuration of the Combined hRBC model. The stretching forces are applied to the highlighted nodes. (Note, this is actually the tenth plot produced by the simulation output, but it is presented first to help illustrate the example simulation itself as well as the degree of deformation experienced by the Combined hRBC model at the conclusion of the simulation).

### hRBC Final Configuration (With Attachment Point Nodes Highlighted)



**FIGURE 4-1:** Visualization of the final configuration of the Combined hRBC model for the example simulation.

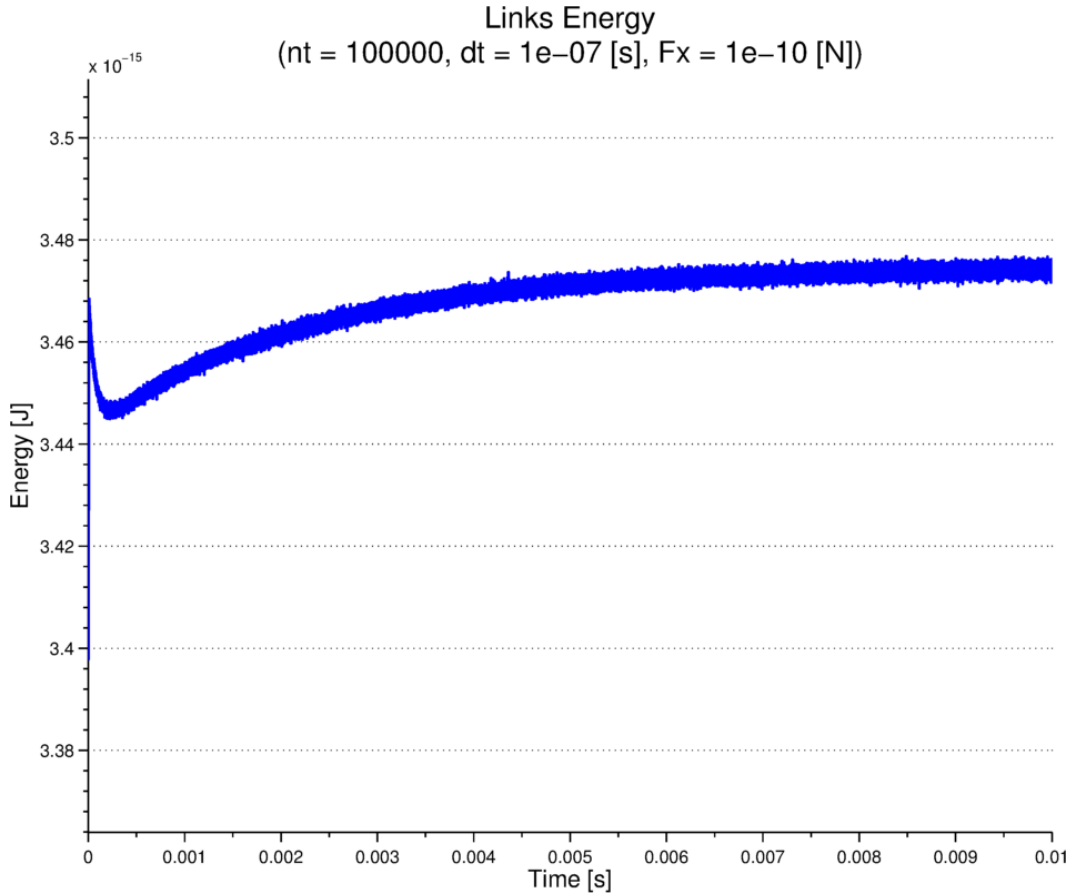
As expected, the model extends in the direction of the stretching forces, while simultaneously contracting in the transverse direction. The deformed model can



be described by an axial diameter (DA) and a transverse diameter (DT).

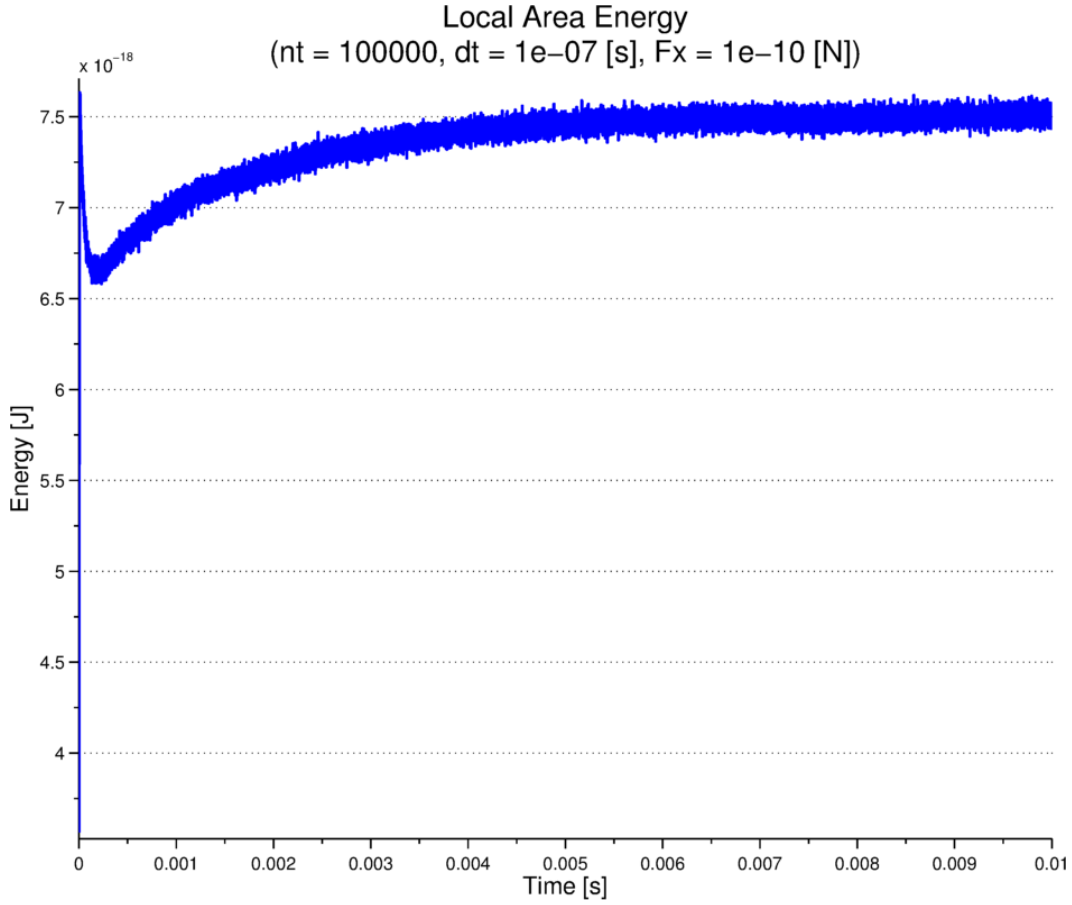
Together, these dimensions can be used to quantify the deformation of the model.

**FIGURE 4-2** is a plot of the potential energy contribution from the spectrin links in the Combined hRBC model as a function of time elapsed in the simulation.



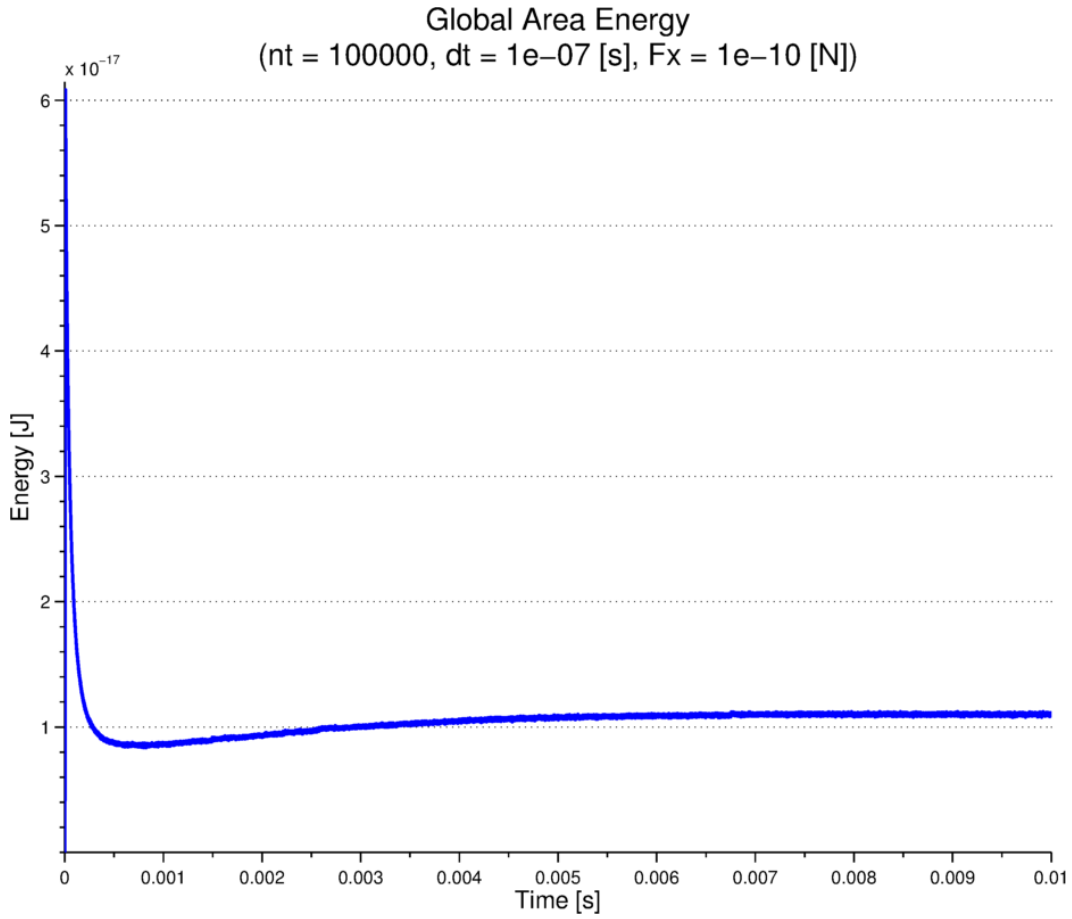
**FIGURE 4-2:** Plot of the contribution to the potential energy from the in-plane term in the force field of the Combined hRBC model as a function of time for the example simulation.

**FIGURE 4-3** is a plot of the potential energy contribution from the areas of the individual elements (local area energy) in the Combined hRBC model as a function of time elapsed in the simulation.



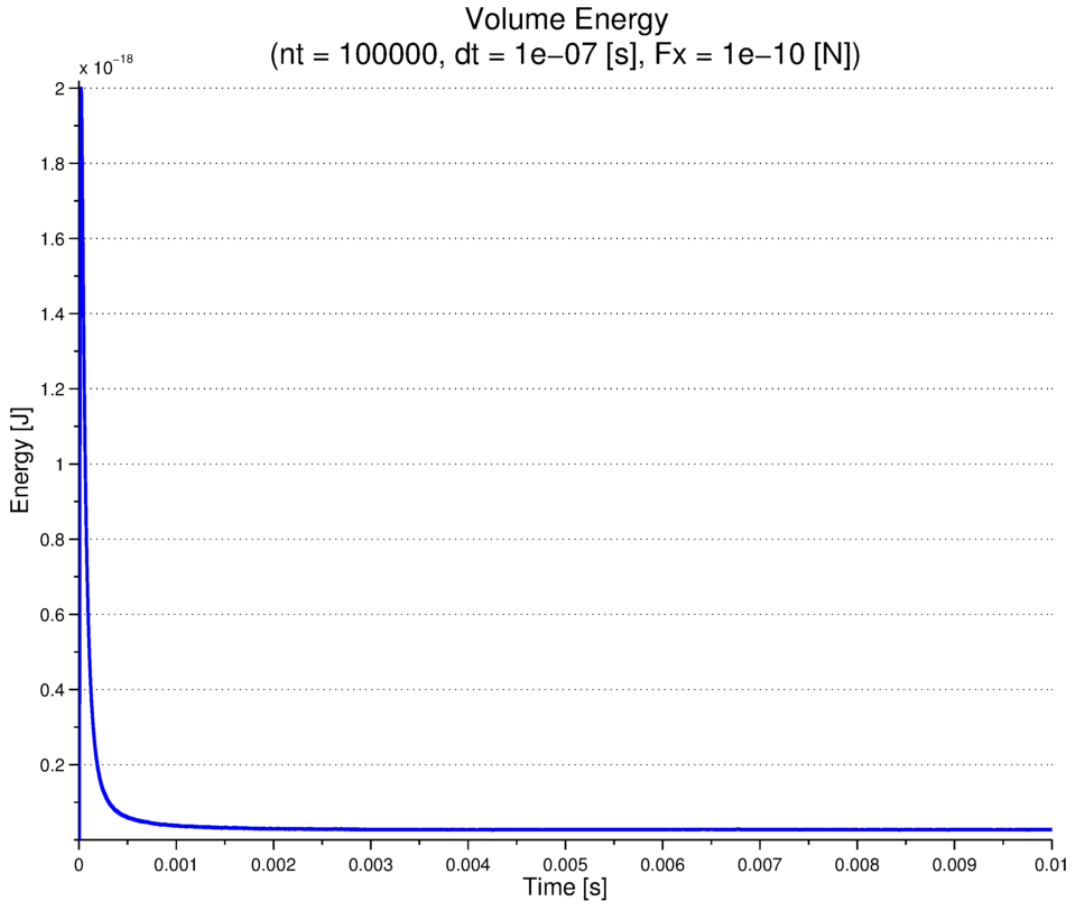
**FIGURE 4-3:** Plot of the contribution to the potential energy from the local area term in the force field of the Combined hRBC model as a function of time for the example simulation.

**FIGURE 4-4** is a plot of the potential energy contribution from the total area (global area energy) of the Combined hRBC model as a function of time elapsed in the simulation.



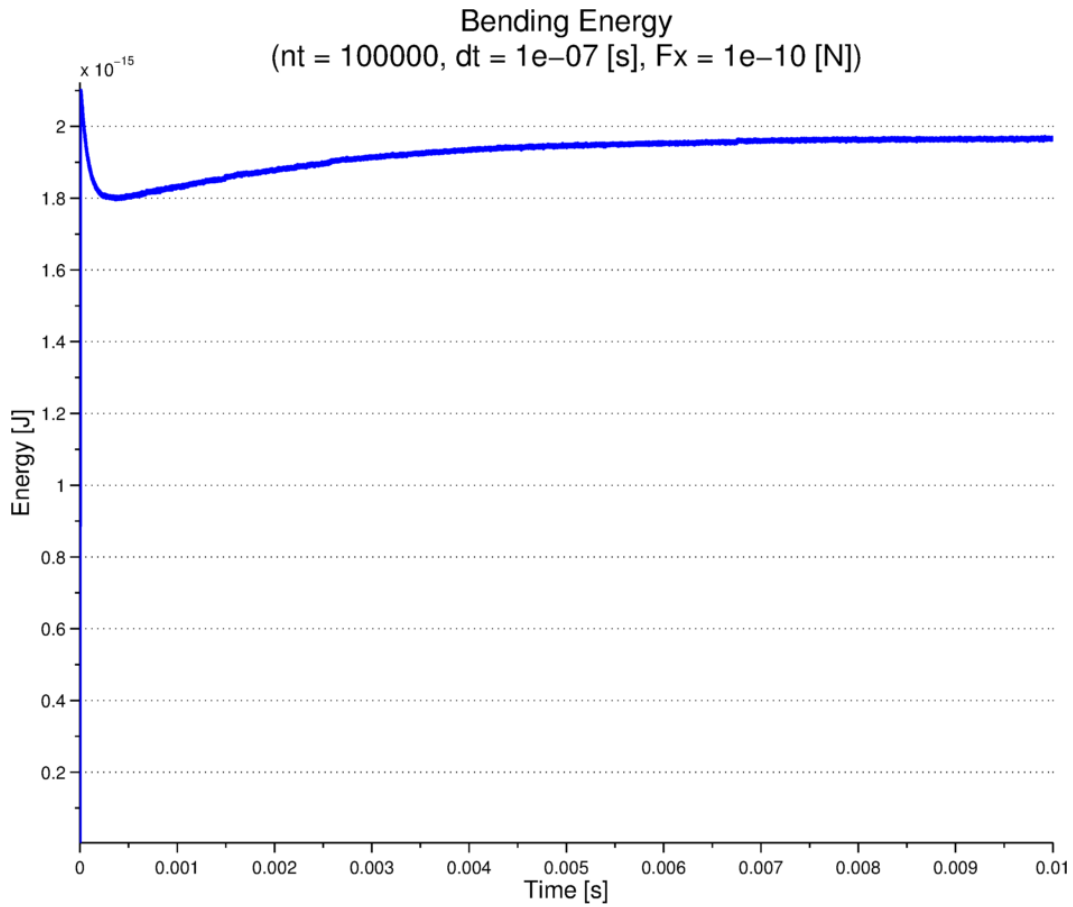
**FIGURE 4-4:** Plot of the contribution to the potential energy from the global area term in the force field of the Combined hRBC model as a function of time for the example simulation.

**FIGURE 4-5** is a plot of the potential energy contribution from the total volume of the Combined hRBC model as a function of time elapsed in the simulation.



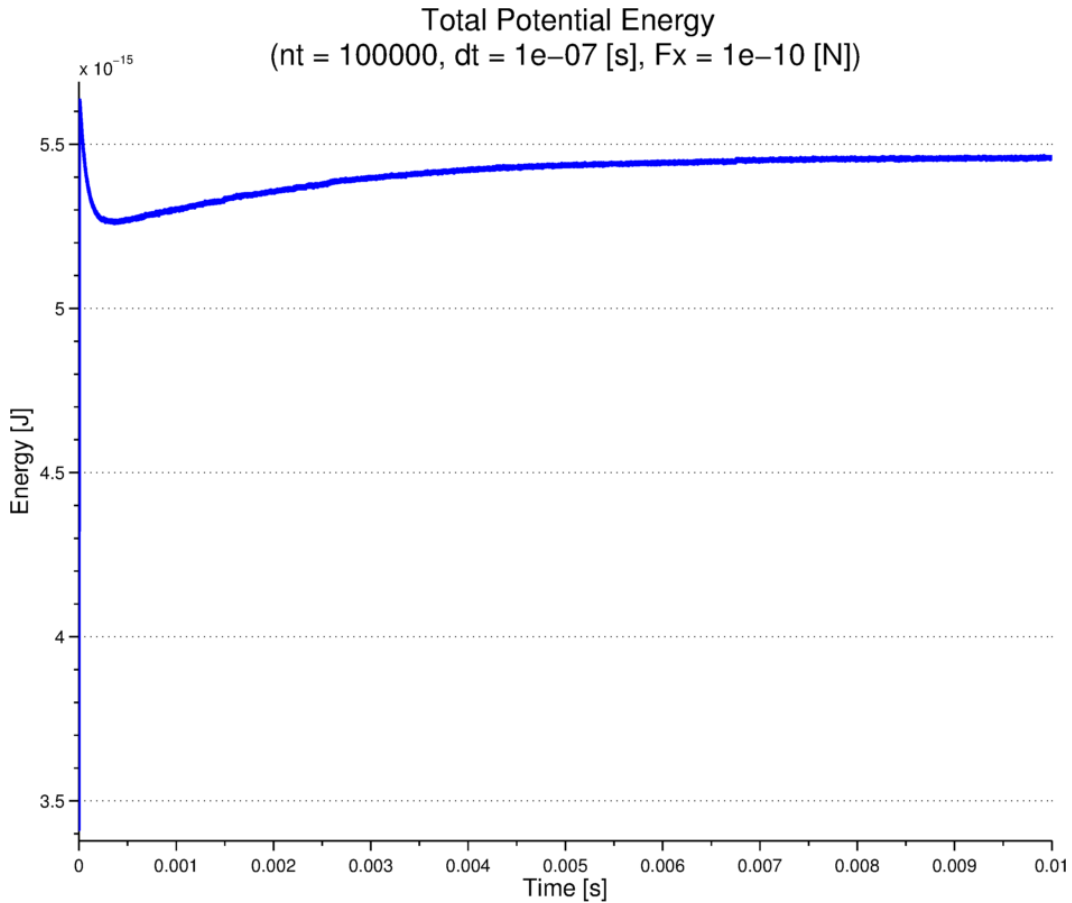
**FIGURE 4-5:** Plot of the contribution to the potential energy from the volume term in the force field of the Combined hRBC model as a function of time for the example simulation.

**FIGURE 4-6** is a plot of the potential energy contribution from the bending between the faces of neighboring elements in the Combined hRBC model as a function of time elapsed in the simulation.



**FIGURE 4-6:** Plot the contribution to the potential energy from the bending term in the force field of the Combined hRBC model as a function of time for the example simulation.

**FIGURE 4-7** is a plot of the total potential energy of the Combined hRBC model as a function of time elapsed in the simulation. From analysis of **FIGURE 4-7**, it can be concluded that the Combined hRBC model had reached potential energy equilibrium at the point when the example simulation ended.

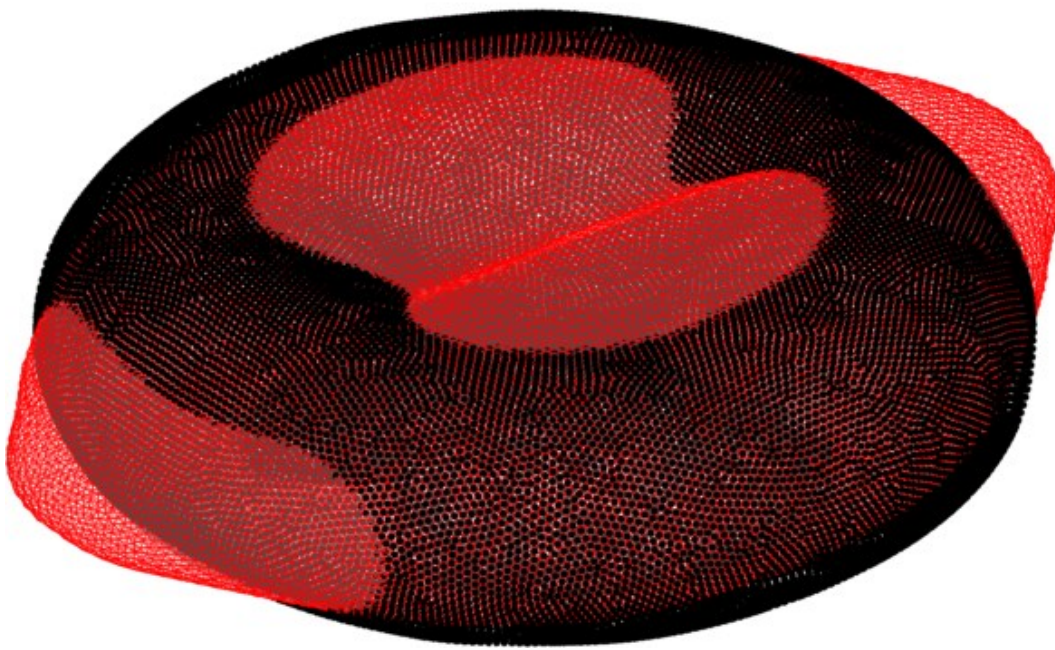


**FIGURE 4-7:** Plot of the total potential energy of the Combined hRBC model as a function of time for the example simulation.

Under inspection, the shape of the total potential energy curve appears qualitatively similar to the shape of the links energy curve (in **FIGURE 4-2**). Under the further investigation, it is apparent that they are equal magnitude. Additionally, the other energies are of lesser magnitudes. From this, it is apparent that the links energy term dominates the other potential energy terms.

**FIGURE 4-8** is a plot comparing the final configuration of the Combined hRBC model to the initial configuration.

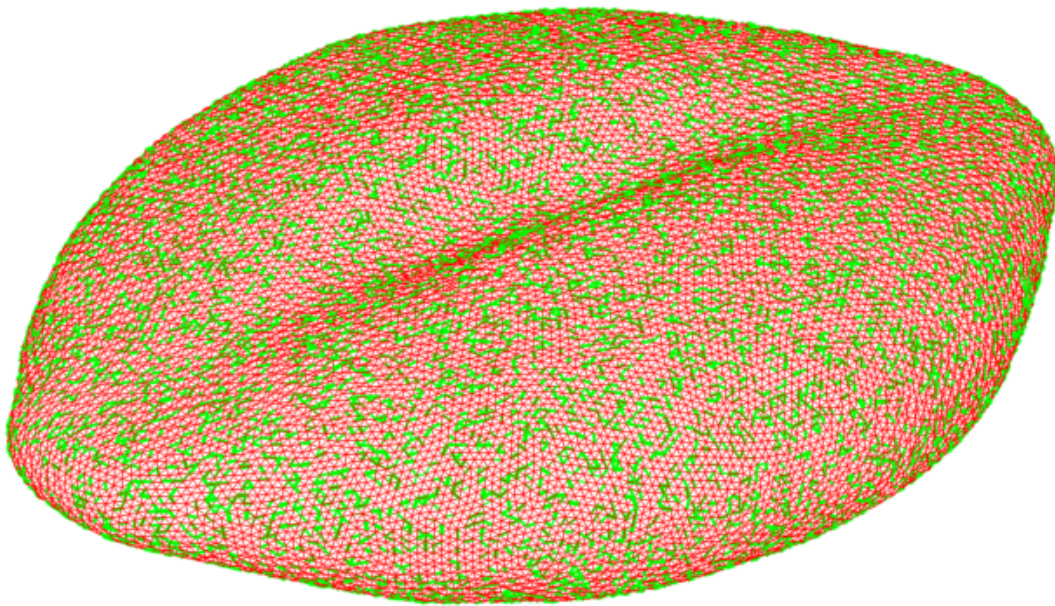
### hRBC Final Configuration (With Initial Configuration)



***FIGURE 4-8:** Visualization comparing the final configuration of the Combined hRBC model (red) to the initial configuration (black) for the example simulation.*

**FIGURE 4-9** is a plot of the final configuration of the Combined hRBC model that highlights the dissociated spectrin links.

### hRBC Final Configuration (With Dissociations)

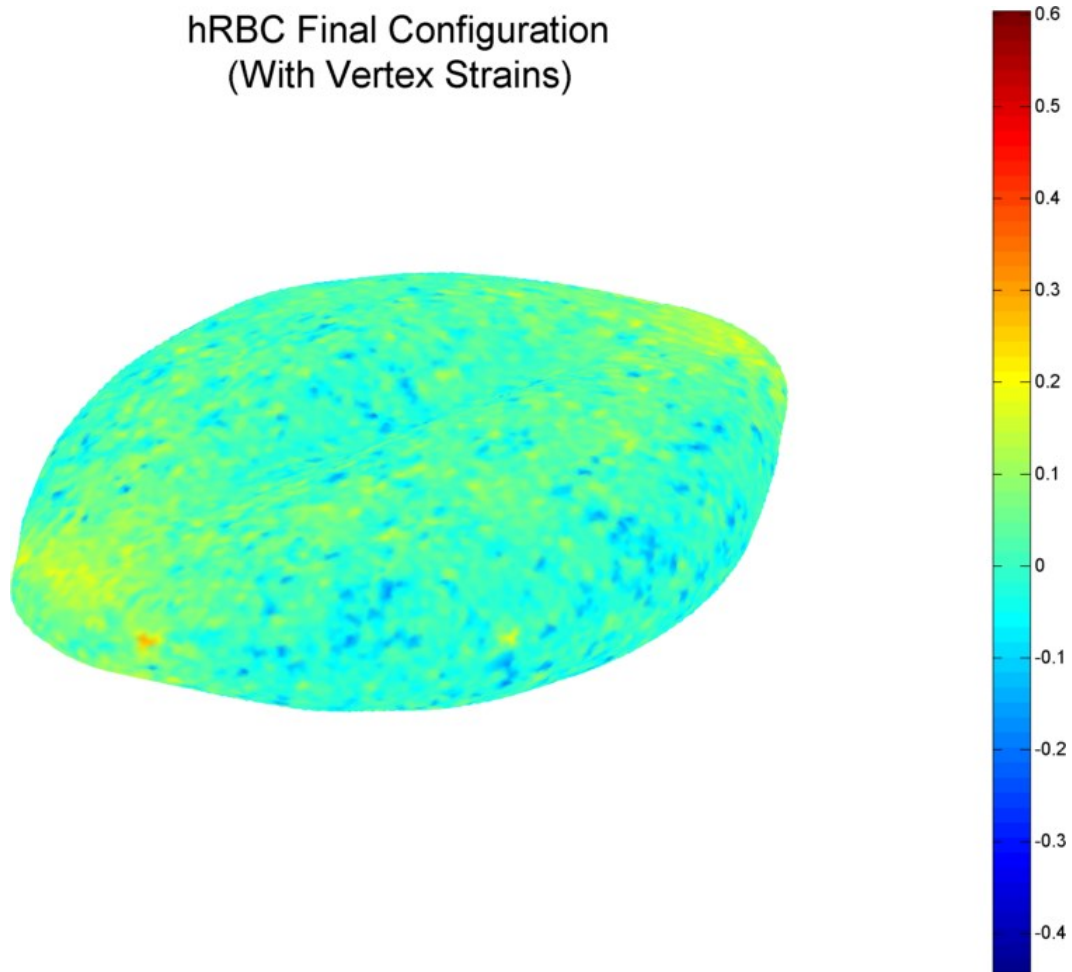


**FIGURE 4-9:** Visualization of the final configuration of the Combined hRBC model with dissociated spectrin links (green) for the example simulation.

From an analysis of the connectivity of the spectrin links in the Combined hRBC model as a function of time elapsed in the simulation, it can be concluded that the Combined hRBC model was not under any type of metabolic loading that would alter its connectivity. It can also be concluded that the forces applied to the Combined hRBC model during the simulation were not large enough to cause any of the spectrin links in the model to break.

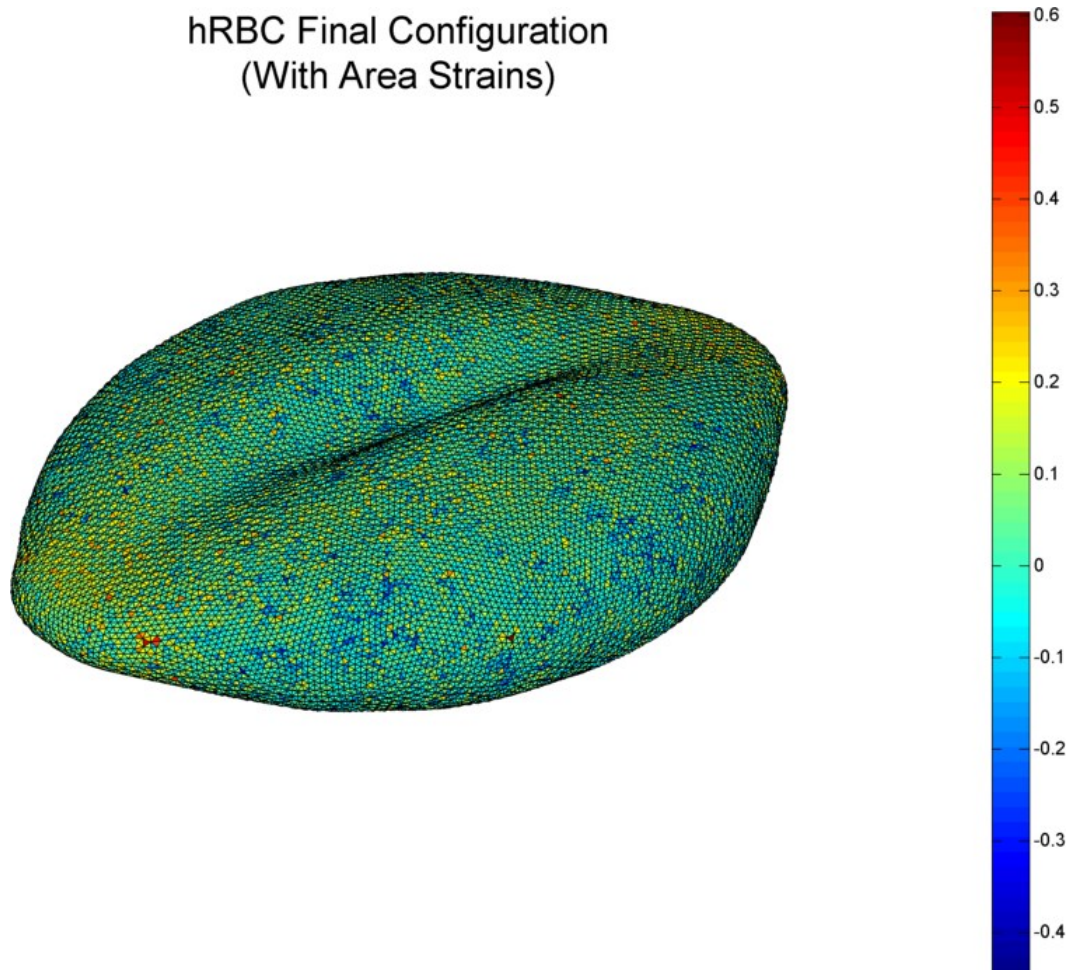


**FIGURE 4-10** is a plot of the final configuration of the Combined hRBC model overlaid with the vertex strains. The vertex strain of a particular vertex is the average of the area strains of the individual elements that share that vertex. From visual analysis of **FIGURE 4-10**, it is apparent that the majority of the elements are in a minimal strain state.



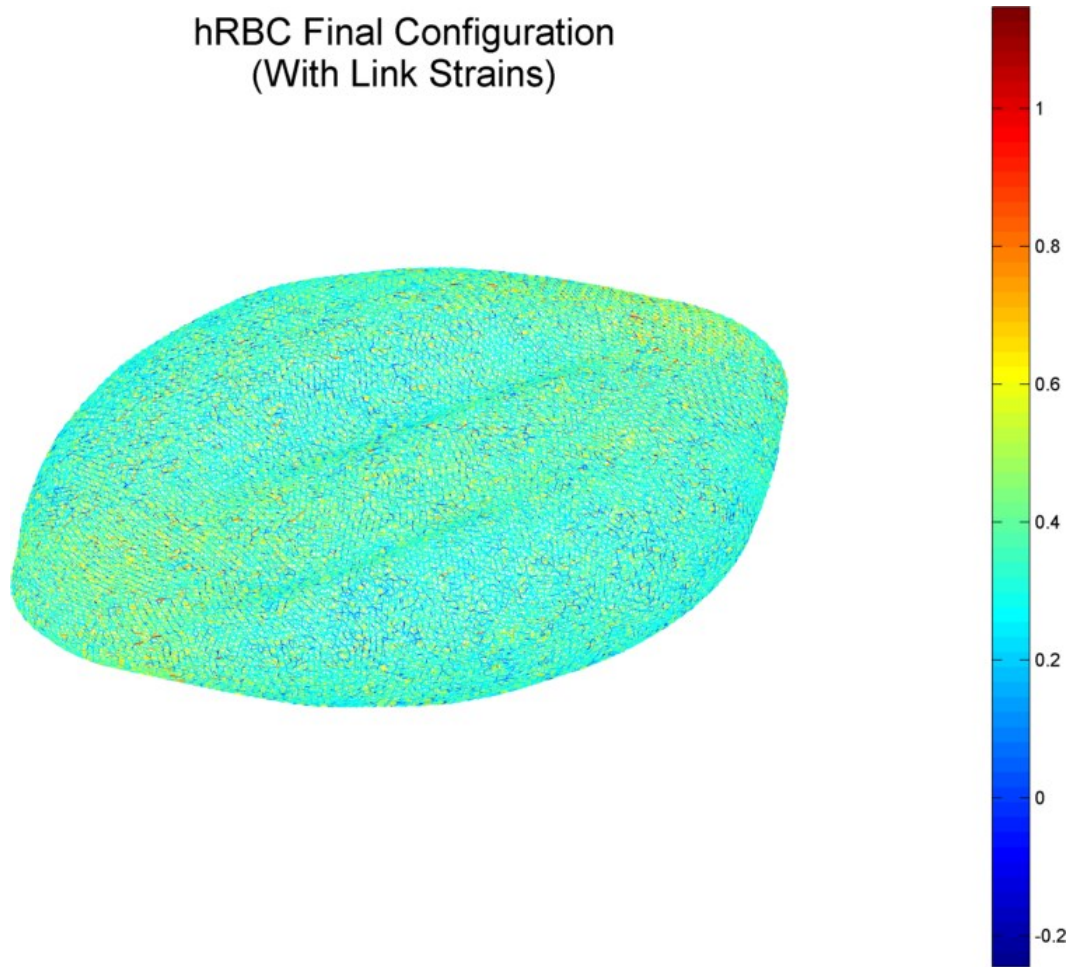
**FIGURE 4-10:** Visualization of the vertex strains in the final configuration of the Combined hRBC model for the example simulation. The colorbar on the right side indicates the magnitude of the strains.

**FIGURE 4-11** is a plot of the final configuration of the Combined hRBC model overlaid with the local area strains (area strains of the individual elements). Areas of higher strain can be observed where the stretching forces are applied.



**FIGURE 4-11:** Visualization of local area strains in the final configuration of the Combined hRBC model for the example simulation. The colorbar on the right side indicates the magnitude of the strains.

**FIGURE 4-12** is a plot of the final configuration of the Combined hRBC model overlaid with the link strains. From visual analysis of **FIGURE 4-12**, the spectrin links with a higher likelihood dissociation can be observed. In this particular case, only a few of the spectrin links seem to have undergone severe deformation.



**FIGURE 4-12:** Visualization of the link strains in the final configuration of the Combined hRBC model for the example simulation. The colorbar on the right side indicates the magnitude of the strains.

In addition to outputting all the required data to produce the above plots, the simulation also outputs:

- Information that can be used to uniquely identify the simulation
  - The starting and ending date and time of the simulation
  - The number of time steps in the simulation
  - The length of a single time step
  - The magnitude of applied force to the Combined hRBC model (if the simulation was a stretching simulation)
- Information about the final configuration of the Combined hRBC model
  - Total area
  - Total volume
  - Axial and transvers diameters
  - Relative shape anisotropy
  - Areas of individual elements
  - Velocities of individual nodes
  - Quality of the triangulation (to determine the deviation from the ideal network assumed in the theory)
- The final metabolite concentrations
- Wall-clock time per time step (to evaluate computational efficiency)

## 4.2 Triangulation Quality

The quality of the triangulation of the Combined hRBC model was evaluated using the metrics that were introduced in [44] to evaluate the quality of the CG-hRBC model. The results from this triangulation quality analysis are tabulated in [TABLE 4-1](#).

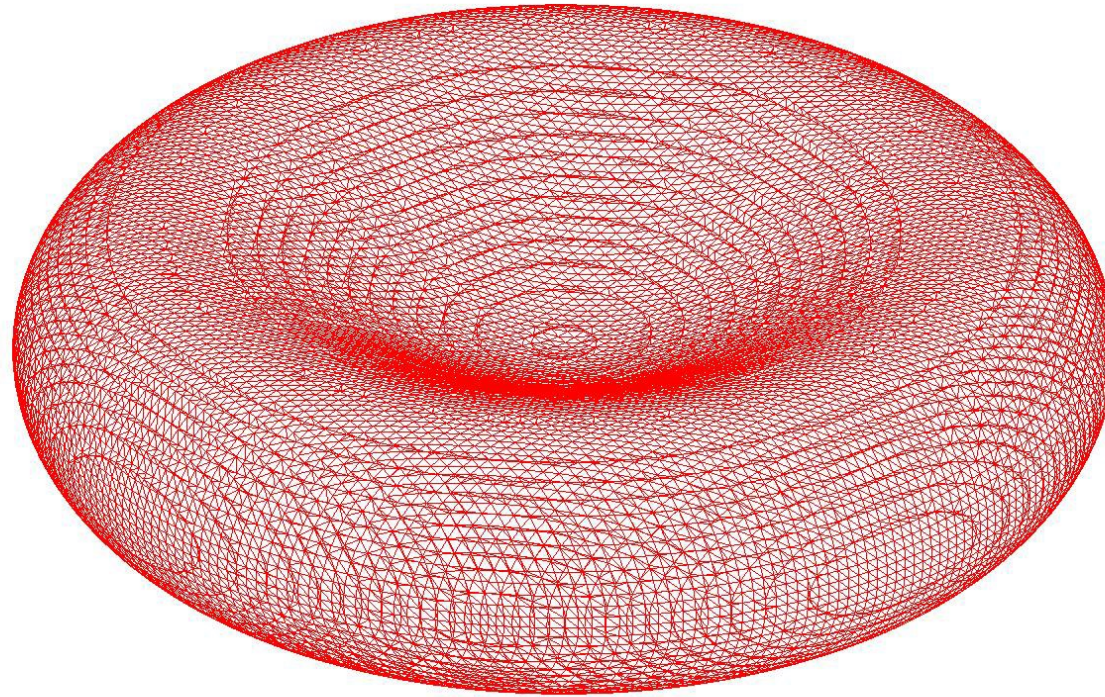
Results from Triangulation Quality Analysis

	Distribution of Link Lengths	Distribution of Degree-6 Vertices	Distribution of Degree-5 and Degree-7 Vertices	Equilateral Triangulation Quality
CG-hRBC Model	0.05 to 0.08	75% to 90%	10% to 25%	-
Combined hRBC Model	0.587	88.89%	11.11%	0.9903

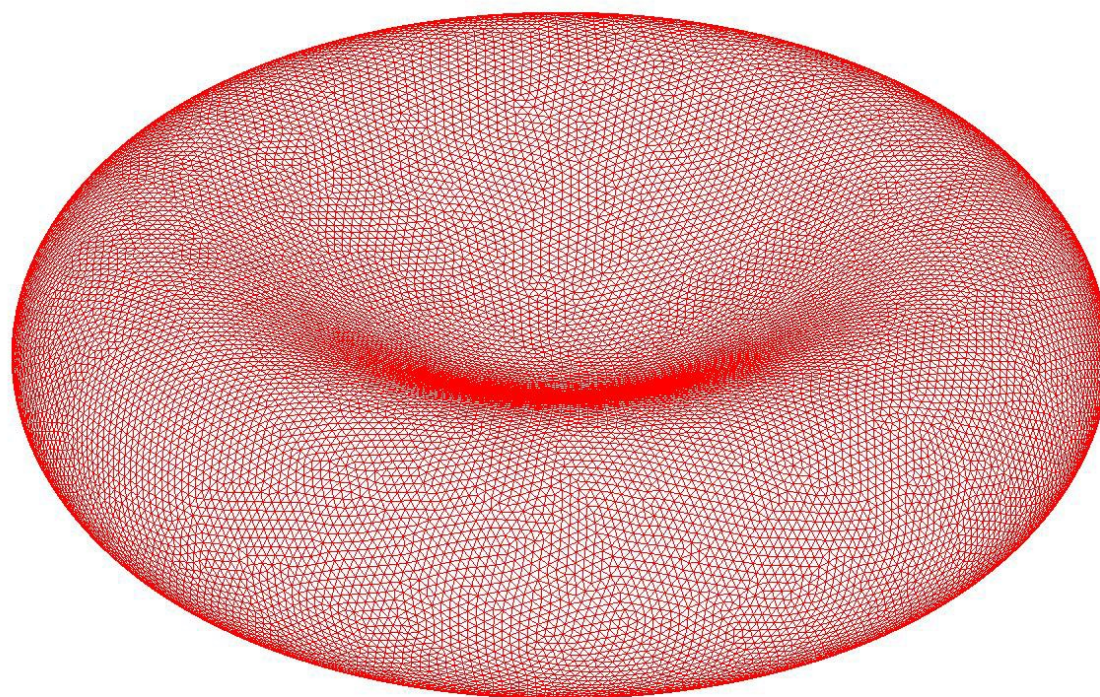
*TABLE 4-1: Results from triangulation quality analysis. Results for CG-hRBC model taken from [44]. The distribution of link lengths of the Combined hRBC model was determined using [EQUATION 2.13](#). The equilateral triangulation quality of the Combined hRBC model was determined using [EQUATION 2.14](#).*

The results from the triangulation quality analysis for both the distribution of link lengths and the distribution of vertex degrees of the Combined hRBC model are near the upper end of the range of the results from the triangulation quality analysis of the CG-hRBC (recall, a lower value for the distribution of link lengths corresponds to a higher quality triangulation). This implies a good correlation between the two models. In addition, the equilateral triangle quality of the hRBC model is 0.9903, with a minimum individual element quality of 0.9009. This implies a good correlation between the Combined hRBC model and the ideal network used in the theoretical analysis. The improvement that the *distmeshsurface.m* function offers over other methods is evidenced by the comparison between [FIGURE 4-13](#) and [FIGURE 4-14](#) (on the next two pages).





*FIGURE 4-13: Visualization of the initial triangulation for the Combined hRBC model, obtained using the isosurface function only.*



*FIGURE 4-14: Visualization of the final triangulation for the Combined hRBC model, after optimization of the triangulation shown in [FIGURE 4-13](#) through use of the `distmeshsurface.m` function.*

The triangulation in **FIGURE 4-13** is the output obtained from just the *isosurface* function. The triangulation in **FIGURE 4-14** is the output from the *distmeshsurface.m* function using the triangulation from **FIGURE 4-13** as input. Clearly, given the quantitative results from the triangulation quality analysis and from qualitative analysis of **FIGURE 4-13** and **FIGURE 4-14**, the usage of the *distmeshsurface.m* function produces a triangulation that is much more regular (the elements are closer to equilateral triangles).

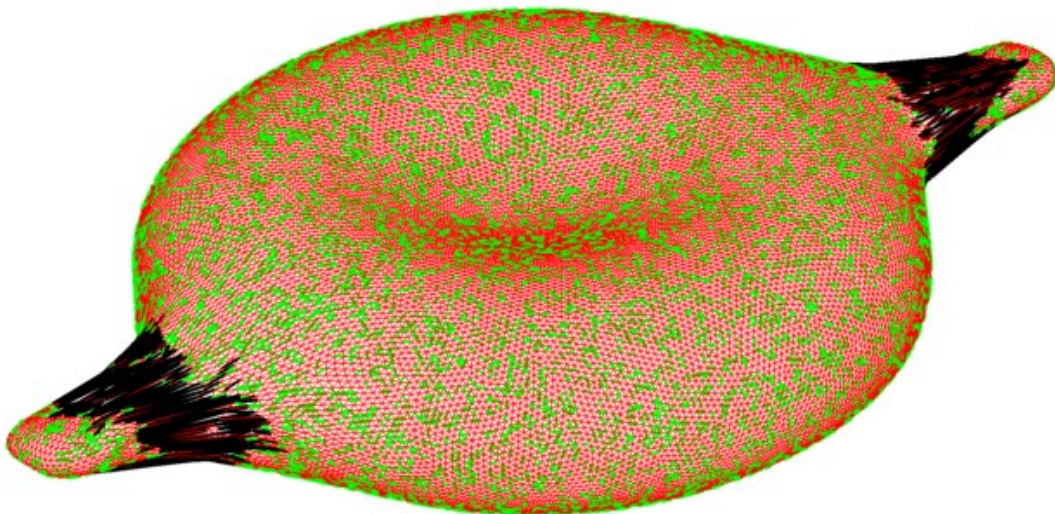
The quality analysis presented above was performed on the hRBC model in its reference configuration, but the same analysis can be performed on the hRBC model at any time during a simulation, regardless of the amount of deformation of the hRBC model. A quality analysis performed on a deformed hRBC model could be used to determine the deviation of the deformed network from the ideal network used in the theoretical analysis.



### 4.3 Breaking of Spectrin Links

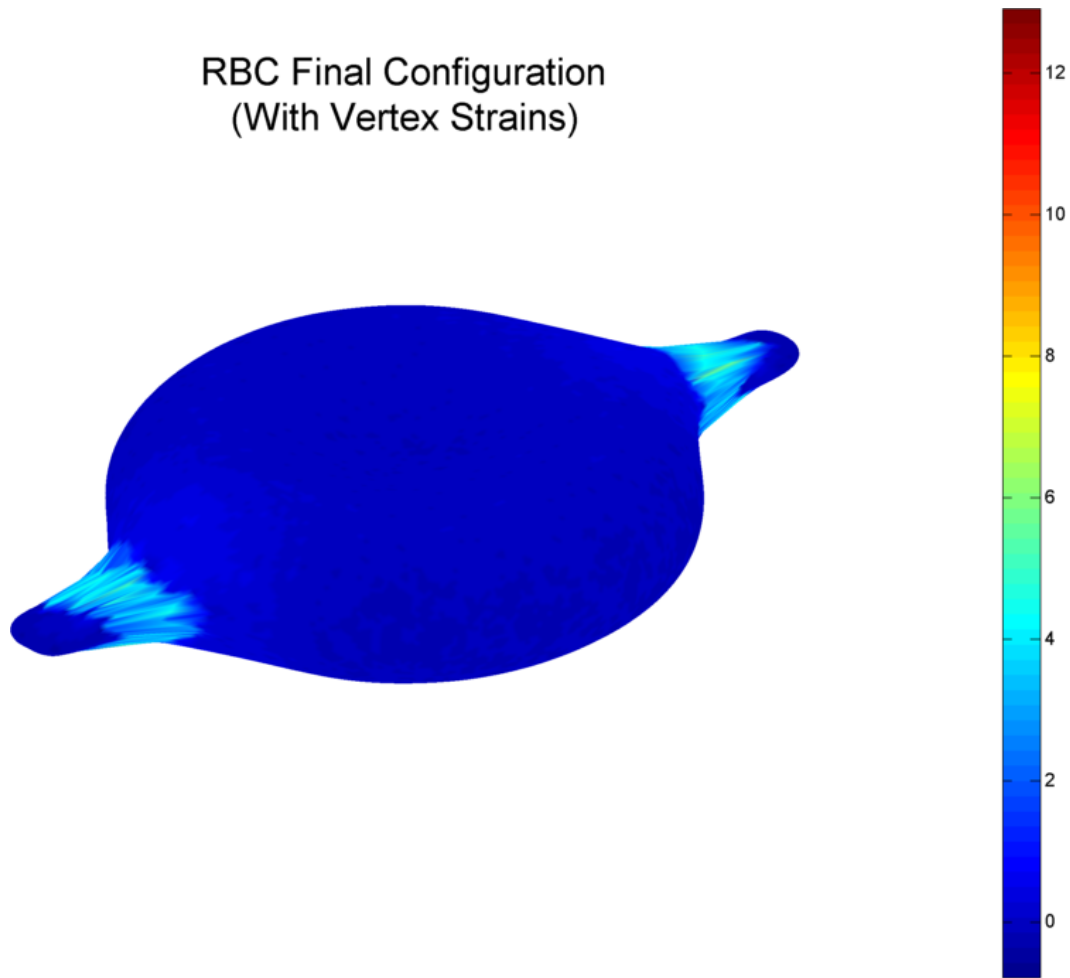
Under the application of a large enough force, breaking of individual spectrin links in the Combined hRBC model can be observed. This behavior corresponds to the extension of the individual links beyond the contour length of the spectrin fiber, as explained in [SUBSECTION 3.2.3](#). The results from a stretching simulation of the Combined hRBC model using an applied force of 1 nanonewton is depicted in [FIGURE 4-15](#). (Note, this force is unrealistic in the context of optical tweezers stretching experiments on the hRBC, but it was used in the simulation simply for illustrative forces).

#### RBC Final Configuration (With Dissociations)



**FIGURE 4-15:** Visualization of the final configuration of the Combined hRBC model with dissociated spectrin links (green) and broken spectrin links (black) for the breaking links simulation.

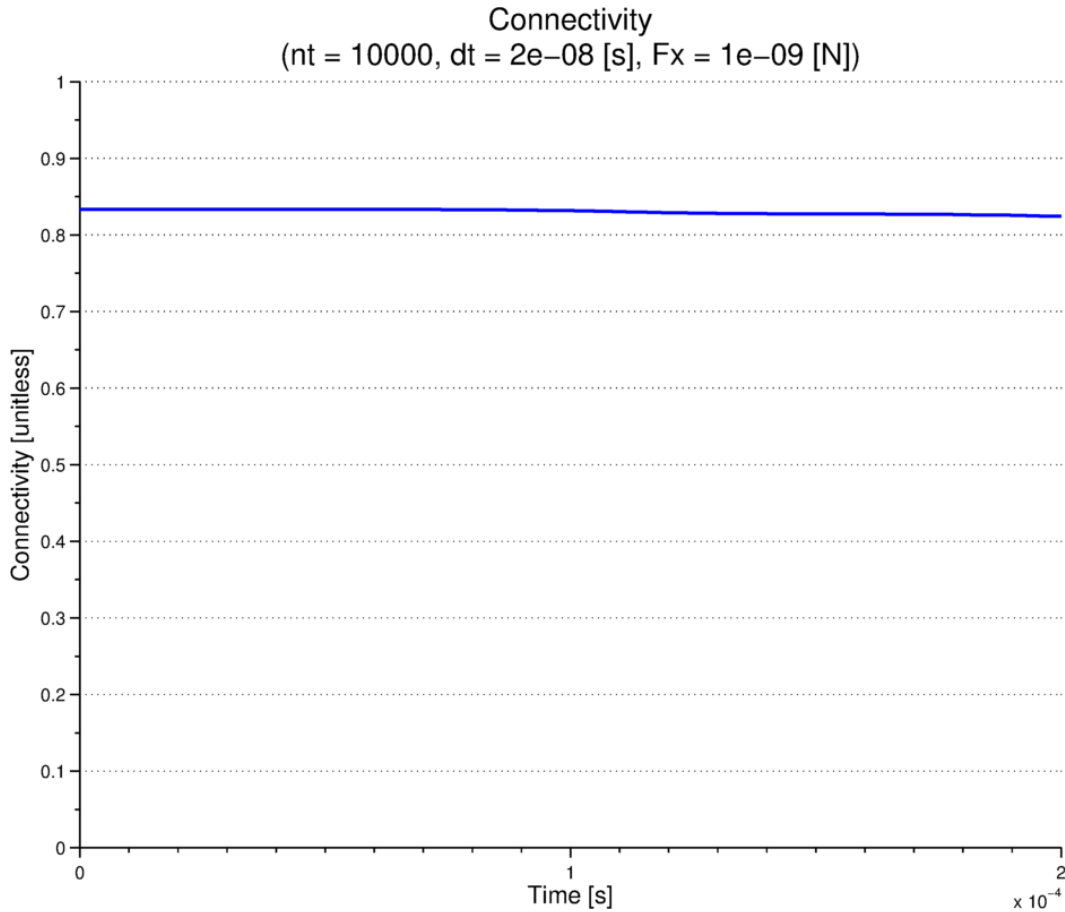
The dissociated links are highlighted in green (same as in the previous stretching simulation), and the broken links in are now highlighted in black. Note how the broken links are concentrated near the areas where the stretching forces are applied to the cell. This results in strain concentrations (and subsequent stress concentrations) in those areas, as shown in [FIGURE 4-16](#).



**FIGURE 4-16:** Visualization of vertex strains in the Combined hRBC model for the breaking links simulation. The colorbar on the right indicated the magnitude of the strains.

Note how the shape of the cell for the breaking links simulation is qualitatively different than the shape of the cell in [FIGURE 4-10](#). This is due to the extremely high stretching force causing the nodes where the force is applied to accelerate outward before the rest of the nodes in the body can respond. The connectivity of

the Combined hRBC model as a function of time is shown in [FIGURE 4-17](#). A slight decrease in connectivity can be observed.



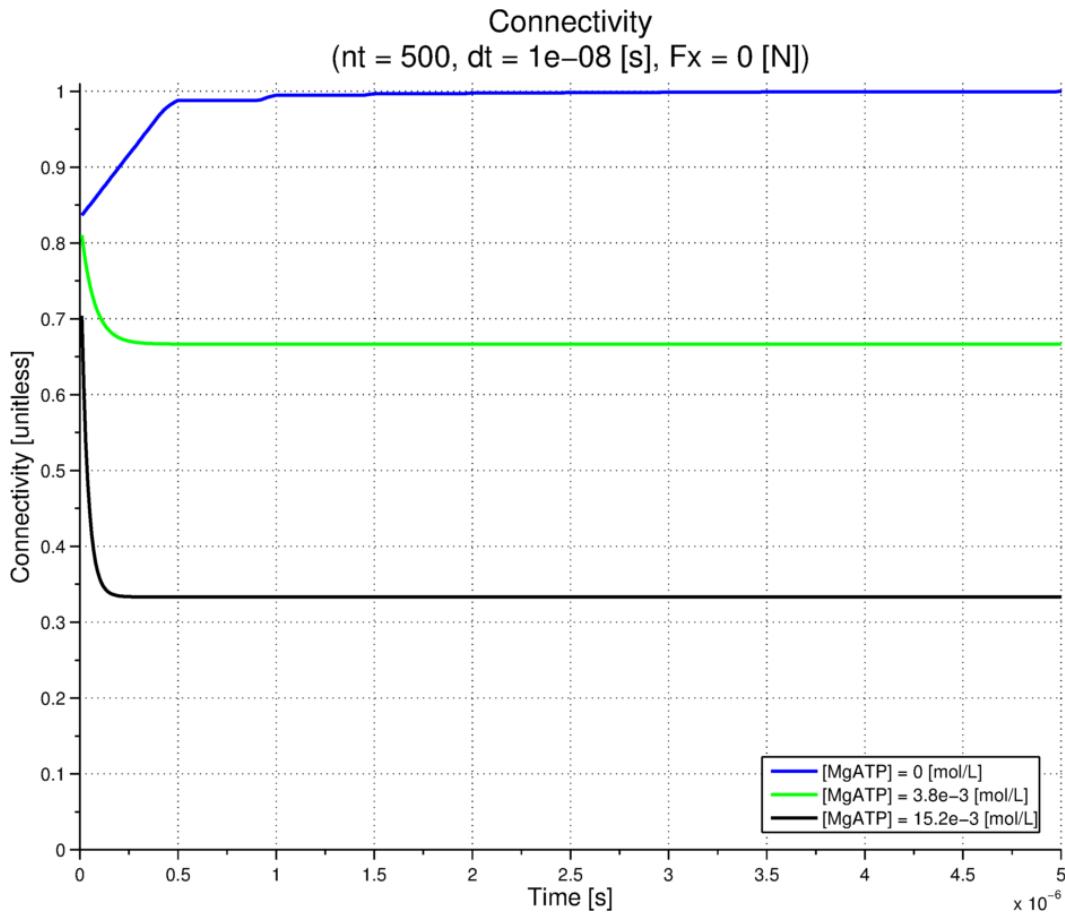
[FIGURE 4-17](#): Plot of the connectivity of the Combined hRBC model as a function of time for the breaking links simulation.

During this simulation, 841 spectrin links are permanently broken as a result of over extension (leaving only 81185 links in the model). The broken links are a small fraction of the total number of links, so the in connectivity seems small in the plot in [FIGURE 4-17](#). The final connectivity of the Combined hRBC model at the end of this simulation of breaking links is still  $p = 5/6$  (67608 associated links). Since the MgATP concentration is at its steady-state value (which dictates that the connectivity of the network should be  $p = 5/6$ ), this indicates that code is operating correctly.

## 4.4 Connectivity as a Function of MgATP Concentration

The dynamic evolution of the connectivity as a function of the MgATP concentration can be observed in simulations of the Combined hRBC model.

**FIGURE 4-18** illustrates the connectivity results from three separate simulations in which the MgATP concentration is artificially held fixed at a critical value for the duration of the simulation. These simulations are analogous to step function inputs to mechanical simulations. In all three simulations, the initial connectivity is  $p = 5/6$ .



**FIGURE 4-18:** Plot of connectivity of Combined hRBC model as a function of time for three cases where the MgATP concentration is held fixed at critical values.

In the first simulation (blue line), the MgATP concentration is held fixed at 0 mol/L. The connectivity increases to a new steady-state value of  $p = 1$  after  $5e-6$  seconds. In the second simulation (green line), the MgATP concentration is held fixed at  $3.8e-3$  mol/L. The connectivity decreases to a new steady-state value of  $p = 2/3$  by  $5e-7$  seconds. This case corresponds to the rigidity percolation threshold of the model network. In the third simulation (black line), the MgATP concentration is held fixed at  $15.2e-3$  [mol/L]. The connectivity decreases to a new steady-state value of  $p = 1/3$ . This case corresponds to the connectivity percolation threshold of the model network. All three steady-state results from the simulations agree with the values predicted by [EQUATION 3.42](#) (shown in [FIGURE 3-7](#)).

The connectivity as a function of time curve is not smooth in the first simulation because the links are reassociating at discrete points in time, according to when their individual wait time counters reach zero. Note that in the second two simulations (where the MgATP is increased from its physiological), the new steady-state connectivities were achieved earlier than in the first simulation (where the MgATP concentration was decreased from its physiological value). This also relates back to the wait time counters restricting the reassociation of the spectrin links.

## 4.5 Discussion and Analysis of hRBC Model

### 4.5.1 Mechanical Behavior

The Combined hRBC model was tested in stretching simulations and was found to be stable under applied forces as large as 200 piconewtons. This value appears to be the largest values used by Fedosov et al. in stretching simulations of the CG-hRBC model [44]. Stability of the model under similar conditions is the first indication that the mechanical portion of the Combined hRBC model (adopted from the CG-hRBC model) can be used to model deformations of the hRBC. Additional simulations are required to compare the behaviors between the two models. Adjustment to the parameters in the Combined hRBC model may be required to ensure that experimentally observed behaviors can be reproduced during simulations.

The response of the hRBC model observed under loading ([SECTION 4.1](#)) can be attributed to the force field used to model the hRBC. Since the WLC-POW-LJ force-extension relationship is linear about the equilibrium length of spectrin extension and the other terms in the force field are harmonic potentials (which means that the forces obtained from these potentials are linear about their equilibrium values), a linear response to infinitesimal strains is expected. The observed strain-stiffening behavior of the hRBC model under finite strains can be attributed to the stiffening of the WLC-POW-LJ force-extension relationship when the extensions are sufficiently larger than the equilibrium length of spectrin extension. The strain-softening behavior of the hRBC model under finite strains can be attributed to the transition in the WLC-POW-LJ force-extension

relationship from the WLC-POW portion to the POW-LJ portion. The force resulting from increasing extension

The use of the modified version of the in-plane force expression (EQUATION 3.22) may result in the observation of “plastic” behavior under large deformations of the network. This can again be attributed to the use of the WLC-POW-LJ force-extension relationship, since the restoring force in the individual link will start to decrease after the link has been stretched beyond a certain extension. This effect on a global scale (network-scale) corresponds to a type of “yielding” of the spring network.

The strain-stiffening behavior of the hRBC observed in experiments [124] is the opposite of the expected behavior of the Combined hRBC model. From the arguments presented earlier, under increasing shear strain conditions one would expect to observe some type of strain-softening behavior, where the hRBC becomes more compliant in order to better fit through the narrow capillaries. This seemingly paradoxical behavior, however, can be explained by examining the nanomechanics of the hRBC cytoskeleton. At large bulk deformations of the hRBC, both the individual links and the network as a whole are significantly deformed from their equilibrium/reference configurations. The force-extension response of the individual links is known to be highly nonlinear [56], so as the network is deformed one would expect to observe some degree of strain-stiffening resulting from the links themselves. In addition, the geometry of the network will be altered from its regular, triangular reference configuration. Under deformation, the links of the network will be more aligned with the direction of

the applied force, which would also result in observed strain-stiffening behavior. Note that this strain-stiffening behavior is strictly in the direction of the applied force. The behavior in other directions may vary differently due to the anisotropic nature of the network.

An interesting, stretching rate-dependent behavior may be observed in simulations. As stated above, large forces applied to the hRBC can cause the spectrin links to break, while small forces will not. If the hRBC is deformed sufficiently quickly, but with a force whose magnitude is not great enough to immediately break links outright, a special case of completely reversible finite deformation may be observed. When the force is applied to the hRBC and it begins deforming, the process of selecting links to dissociate is shifted in favor of the more stretched links. When links are dissociated, the restoring force (between the actin nodes they were previously connected to) is reduced and the separation distance between the nodes can increase more easily. If the stretching rate is sufficiently large, the separation distance between actin nodes may increase to a value greater than the contour length of the spectrin filament before the wait time on the reassociation of the spectrin link is up. In this case, separation distances between actin nodes greater than the contour length are achieved without permanent breaking of the spectrin link. This may allow the hRBC to undergo the large deformations observed in circulation and return to its equilibrium, biconcave discoid shape when the forces causing the deformation are removed. This is speculation about the behavior of the hRBC model, however. Whether or not this corresponds to behavior of the actual hRBC is unknown.



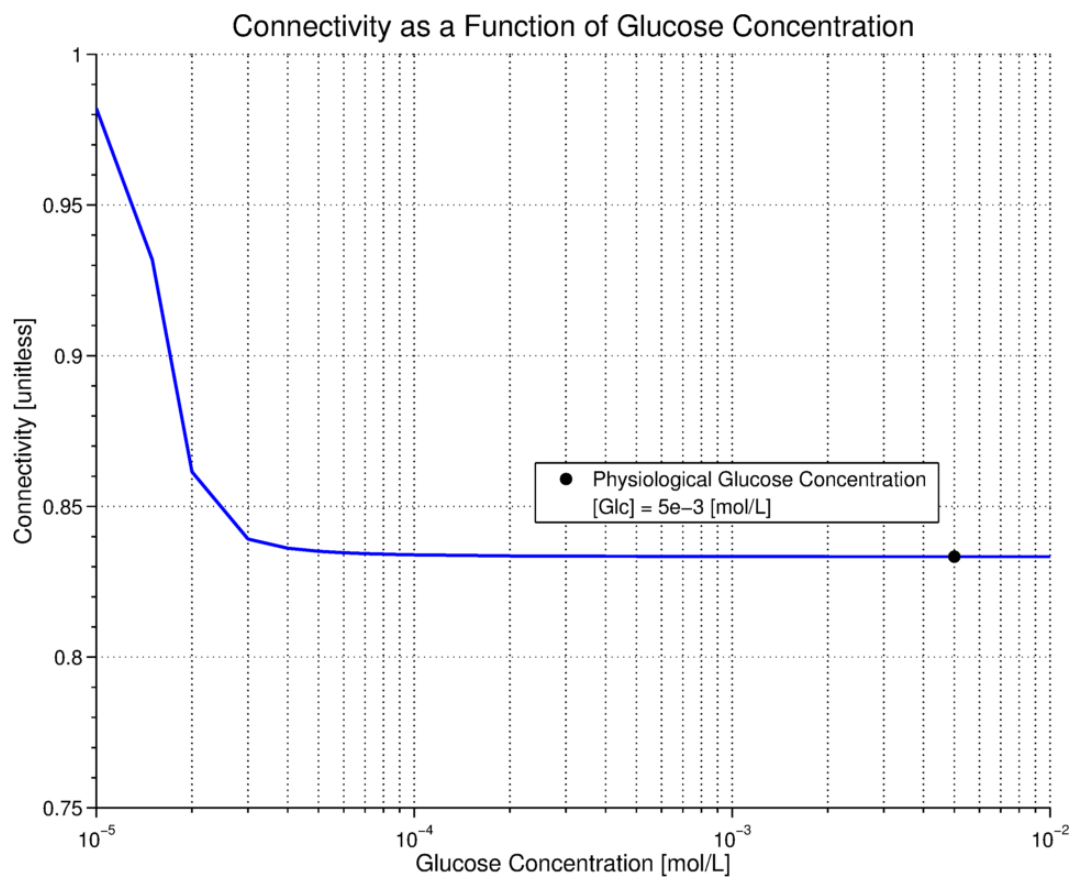
More stretching simulations of the Combined hRBC model are required to fully evaluate the behavior of the model. These stretching simulations should calculate the elongation index for the hRBC model with two different sets of parameters. The first simulation should use the set of parameters from the original CG-hRBC model (with no metabolic simulation). The second simulation should use the set of parameters from the Combined hRBC model (with metabolic simulation). Stretching simulations performed using the Combined hRBC model (with its metabolism at steady-state, so that the connectivity of the network is  $p = 5/6$ ) should have the same results as the same simulations performed on the CG-hRBC model. This test could be used to evaluate if the MFT assumptions made about the shear modulus being a function of the connectivity is valid (and whether the equations used to relate the shear modulus to the connectivity are accurate).

Although the WLC-POW-LJ force-extension relationship is continuous, the use of the Heaviside function in the functional form of the relationship to switch between the WLC-POW portion and the POW-LJ portion causes discontinuities in the integral and derivative of the relationship at the switching point. With proper matching of integration constants, the integral (energy-extension relationship) can be made continuous, but the derivative (stiffness-extension relationship) cannot be made continuous. This undesirable discontinuity could be eliminated if a continuous approximation of the Heaviside function were used (such as modified logistic function).

### 4.5.2 Metabolic Behavior

Using the provided initial conditions [15], the metabolic model was able to reach steady-state within 100,000 seconds of simulation time. Some adjustments to the parameters in the model were required after the dissociation and reassociation reactions for the spectrin links were added to the MMwM model in order to ensure that the proper steady-state connectivity was achieved. Further adjustments to the metabolic parameters may be required to ensure that the metabolite concentrations are accurately reproduced during simulations.

An analysis of the connectivity of the Combined hRBC model as a function of the glucose concentration was also performed. In this analysis, a series of simulations were performed using different values for the glucose concentration (fixed value). Each simulation was run until the metabolism reaches steady-state. The results from this analysis are plotted in [FIGURE 4-19](#). From these results, it can be determined that the connectivity of the Combined hRBC model is unaffected by increases in the glucose concentration to concentrations greater than its physiological concentration ( $5\text{e-}3$  mol/L). The connectivity is also relatively insensitive to changes to the glucose concentration around its physiological concentration. A decrease in the glucose concentration from  $5\text{e-}3$  mol/L to  $5\text{e-}5$  mol/L results in a connectivity change from 0.83 to  $\sim 0.84$ . The connectivity is highly sensitive, however, to changes in glucose concentration when the glucose concentration is  $\sim 10\%$  of its physiological value. A decrease in the glucose concentration from  $5\text{e-}4$  mol/L to  $1\text{e-}4$  mol/L results in a connectivity change from  $\sim 0.84$  to  $\sim 0.98$ .



*FIGURE 4-19: Plot of connectivity of Combined hRBC model as a function of glucose concentration on a semilog plot.*

The insensitivity of the connectivity to increases in the glucose concentration is most likely due to limitations in the maximum rates of some of the reactions in the metabolism, since the MgATP concentration does not increase with the glucose concentration in these cases.

### 4.5.3 Mechanical-Metabolic Behavior

A state of dynamic equilibrium can be observed when the hRBC is at rest in its reference configuration and the metabolism is at steady-state. In this state, the connectivity of the Combined hRBC model remains constant at  $p = 5/6$ . Dissociations and reassociations of spectrin links are occurring every time step, but in equal amounts. This indicates that the coupling between the two models is working correctly. However, the observed equilibrium state is the result of several assumptions that were required in the derivation process due to the lack of experimental data on the dynamic dissociation-reassociation of the hRBC cytoskeleton. Experiments on the hRBC, specifically on the interaction between the metabolism and the cytoskeleton, would help to validate these assumptions.

### 4.5.4 Stress-Free Condition

It was originally thought that the use of the stress-free condition may be necessary if the observed shape memory behavior of the actual hRBC [125] was to be reproduced in simulations. In the stress-free condition, each spectrin link in the model has unique parameters, chosen to eliminate stress concentration in the cytoskeletal network resulting from an imperfect triangulation [80]. A network consisting of unique springs is thought to have a single reference configuration, which means that hRBC models may be able to reproduce the shape memory behavior. The use of the stress-free condition [44] in the Combined hRBC model, where, active dissociation and reassociation of links are taking place during simulations, may not be feasible for two reasons.

First, from a theoretical standpoint, the active dissociation and reassociation of the links can have a similar purpose as enforcing the stress-free condition, which is to help eliminate stress concentrations in the computational membrane that result from an imperfect triangulation. The active dissociation and reassociation of links is weighted towards eliminating tensile stress concentrations in the membrane, resulting from individual links that are extended a distance greater than their equilibrium lengths, while the stress-free condition eliminates both tensile and compressive concentrations by rescaling the equilibrium lengths of individual links until no stress concentrations exist anywhere in the membrane (computational annealing). Second, from a computational standpoint, it would be difficult to enforce the stress-free condition because the active dissociation and reassociation of links is a dynamic process and link lengths are constantly changing, even when the model hRBC is at a potential energy minimum. The shape memory behavior may be able to reproduce through other means, as discussed in [SUBSECTION 5.2.1](#).

#### 4.5.5 Mass Scaling Discrepancy

The simulations performed Fedosov et al. [44, 93, 94] scale the CG-hRBC model so that the simulation form has non-dimensional units. Some of the parameters Fedosov et al. provide are only in the non-dimensional form. For these parameters to be used in simulations in this thesis, they needed to be converted back into their dimensional form using the dimensional scaling relations provided by Fedosov et al. One of the parameters presented without a dimensional equivalent is the nodal mass (mass of the hRBC per node) of the CG-hRBC. Fedosov et al. assume that the non-dimensional mass is equal to 1. This assumption, however, is four to five orders of magnitude greater than the actual nodal mass. The mass scaling discrepancy results are tabulated in [TABLE 4-2](#).

Results from Mass Scaling Discrepancy Analysis

		Non-dimensional	Dimensional
CG-hRBC Model	Total Mass:	27344	2.34e-9 [kg]
			1.71e-8 [kg]
	Nodal Mass:	1	8.56e-14 [kg/node]
			6.24e-13 [kg/node]
Combined hRBC Model	Total Mass:	1.25	1.07e-13 [kg]
		0.17	
	Nodal Mass:	4.59e-5	3.93e-18 [kg/node]
		6.30e-6	

**TABLE 4-2:** Results from mass scaling discrepancy analysis. hRBC density was taken to be  $\rho_{rbc} = 1.15\rho_{water}$ .

An error in the mass would not affect the results from static stretching test, but would result in errors in dynamic tests (or any time-dependent test). An error of this magnitude may cause the simulation to be unstable under any conditions.

#### 4.5.6 Dissipative Forces Error

It is the belief of this author that the use of the fluid particle model [98] is invalid at the length scales of interest for hRBC modeling. The fluid particle model is a generalization of dissipative particle dynamics (DPD) [97]. It attempts to balance the dissipative and random (due to thermal fluctuations) forces acting on a particle suspended in a fluid according to the fluctuation-dissipation theorem. The fluid particle model is used in simulations of the CG-hRBC model to account for the viscosity of the cell membrane, cytoplasm, and blood plasma by applying dissipative forces (functions of nodal velocities) and random forces to the nodes in the model.

When the dissipative forces derived from the fluid particle model were applied to the nodes of the Combined hRBC model in simulations, an extremely small time step length (on the order of 1 femtosecond) was required to mitigate the instabilities that were introduced. Specifically, the magnitudes of the dissipative forces were several orders of magnitude greater than the magnitudes of the conservative forces. This resulted in the dissipative forces dominating the simulations. This in turn resulted in the simulations diverging and failing to complete due to errors in the in the calculations of the conservative force expressions (due to the fact that the nodes of the hRBC were now many meters apart).

This instability introduced by the dissipative forces may be due, in part, to the usage of the incorrect mass in the CG-hRBC model ([SUBSECTION 4.5.5](#)).

When the new dissipative force expressions are used ([SUBSECTION 3.2.3](#)), a more

reasonable time step length (as large as 25 nanoseconds) can be used without causing the simulation to diverge. It is important to note that the use of dissipative force in purely stretching simulations is unnecessary, since only the final configuration of the model is of interest.

Despite the reformulation of the viscous contribution from the membrane to the behavior of the hRBC ([EQUATION 3.29](#) and [EQUATION 3.30](#)), unrealistically large dissipative forces may still be obtained during simulations. The nature of this error is currently unknown (since the dissipative force is based off of the measured viscosity of the hRBC membrane), but the effect can be negated somewhat in order to prevent the simulations from diverging. After the calculation of the dissipative forces for each node, the magnitude of the dissipative force for each node is calculated and compared to the magnitude of the conservative force acting on that node (total conservative force, include contributions from in-plane, local area, global area, bending, and volume). If the magnitude of the dissipative force acting on a particular node is greater than the magnitude of the conservative force acting on that node, the magnitude of the conservative force is scaled back so that it is equal to the magnitude of the conservative force. This method, while not perfect, maintains the directionality of the dissipative force. Improvement upon the modeling of the viscoelasticity of the hRBC would be a huge step forward in terms of the accuracy of the Combined hRBC model, as discussed in [SUBSECTION 5.2.1](#).



## 4.6 Computational Efficiency Results

Computational analysis of the code used to simulate the Combined hRBC was initially not a research objective, but it became apparent very quickly that computational efficiency was going to be a major concern if simulation results were to be obtained in a reasonable time period. The wall time for a single time step in early versions of the mechanical model simulation code (performing potential energy minimizations mechanical only) was on the order of 5 to 10 seconds.

This computational inefficiency was also observed in simulations of the metabolism. Early versions of the metabolic model simulation code used a simple Euler method to numerically integrate the system of equations. The wall time per time step was small (only ~6 microseconds per time step), but an enormous number of time steps were necessary to reach steady-state because a small time step length was required (20 nanoseconds of simulation time per step). Using 100,000 seconds of simulation time (which is an order of magnitude less than the simulation time suggested in by the authors in [15]) as the condition to ensure that steady-state is reached requires  $5 \times 10^{12}$  time steps, which equates to 30,000,000 seconds (~347 days) of wall time.

Most of the non-built-in functions used in this thesis are coded as MEX-files (MATLAB® Executable-file). These functions were originally coded as MATLAB® function files, and then converted to MEX-files using the MATLAB® Coder. The conversion of the functions to MEX-files resulted in significant gains in computation speed.

After optimization, the wall time for a single time step in a simulation is ~0.2 seconds. In general, the simulations performed as part of this thesis ranged from  $1e5$  to  $1e6$  time steps, resulting in a simulation wall time between ~5.8 and ~55.5 hours.

All simulations were performed locally on an Intel® Core™ i5-2400 CPU operating at 3.10 GHz with 4.00 GB of RAM running MATLAB® 2012a.

## Chapter 5: Conclusions and Future Outlook

### 5.1 Intellectual Contributions, Anticipated Benefits, and Progress on Research Objectives

A Combined hRBC model was created. This model mode integrates the MMwM model of the hRBC metabolism into the CG-hRBC model of the mechanical behavior of the hRBC and couples them through the connectivity of the cytoskeleton. Originally, a slightly different model of the hRBC was envisioned as the goal for this thesis. This model, while still a combined mechanical-metabolic model, would also account for the effects of physically loading the hRBC (such as in blood flow) on the metabolism. The idea was that the cell would operate as a pseudo-control system with biochemical feedback. This idea represents a baseline for a complete model of the hRBC. Possible methods for the creation of such a model are discussed in [SUBSECTION 5.2.2](#).

A simulation environment for the Combined hRBC model was also created in MATLAB®. This environment is described in detail in the body of the thesis, and the relevant code files are included in [APPENDIX A.1](#) of this thesis.

It will take some time after this thesis is published for its impact on the interest in modeling and simulation of the hRBC to be determined. To the best knowledge of the author, the Combined hRBC model presented in this thesis represents the only coarse-grained MD model of a hRBC with a cytoskeleton that has dynamic connectivity, controlled through a concurrent simulation of the hRBC metabolism.

## 5.2 Potential Future Work

There are a multitude of directions future work could take that would build upon the work performed as part of this thesis. A number of possible future projects have been grouped according to similar end goal, and are described in some detail in the following subsections.

### 5.2.1 Accuracy of Model

The primary goal of this thesis was to improve the accuracy of current hRBC models with respect to reproducing the behavior of an actual hRBC. There are numerous areas where the accuracy of the current model can be improved. In general, the greatest improvements in the accuracy of the Combined hRBC model would be gained through additional experimentation upon the cell. This would provide more accurate information regarding the response of the hRBC under a variety of conditions, which could then be used to update the parameters in the model.

In addition to updating the parameters themselves, incorporating more accurate expressions for the interaction potentials in the force field will improve simulation accuracy. The in-plane, two body interaction between topologically connected nodes (the WLC-POW force-extension relationship) is relatively accurate, since the expression for attraction is was obtained from force-extension tests of individual polymers [56]. Unfortunately, the WLC model cannot fully reproduce all the behaviors of spectrin as a polymer chain. One of the assumptions made in the derivation of the WLC model is that the polymer chain is structure-less. Stretching experiments on single spectrin chains, however,

produce results with complex behaviors that are not possible if the spectrin chain is structure-less [31]. The most significant of these is the observed folding-unfolding behavior of spectrin [126]. More accurate models of the force-extension response of the spectrin chain could be determined from stretching experiments on single chains, alternative models [99, 100] and/or simulations of individual chains [127, 128].

The POW term in the WLC-POW force-extension relationship is intended to account for all repulsive effects between actin nodes connected by spectrin links. This includes steric repulsion between the actin nodes, the repulsion experienced from severely bending a section of a relatively stiff polymer, and the repulsion experienced from severely bending a local portion of the cell membrane. The form of the POW term is too simple to account for the complex nature of the interactions between actin nodes connected by spectrin links that result in the observed repulsive force. In addition, it does not account for interactions between actin nodes not connected by spectrin links, which should also experience repulsion (from the same sources as connected actin nodes, although to less of a degree). By treating the lipid bilayer as a bend elastic cylindrical shell, Li et al. [101] were able to incorporate a repulsive force between connected actin nodes that had a physical basis in the bending of the lipid bilayer.

The bending of the lipid bilayer is already incorporated into the force field of the Combined hRBC model. However, the bending term in the force field accounts for four body interactions between the nodes in neighboring elements, not two body interactions between actin nodes connected by spectrin links. The

discrepancy between the effects of the two body interactions and the four body interactions decreases as the number of nodes in the Combined hRBC model increases, but it still needs to be considered in the interest of completeness. Replacing the POW term in the WLC-POW force-extension relationship with a repulsive force term based upon the bending of the lipid bilayer may improve the accuracy of the Combined hRBC model.

The use of the LJ force-extension relationship in the expression for the in-plane forces in the force field allows for dynamic dissociation and reassociation of the links as a function of the stretching of the link. As detailed in [SUBSECTION 3.2.3](#), this modification was inspired by the cytoskeletal dynamics model proposed by Li et al. [101]. Since the spectrin chain is not explicitly modeled in the Combined hRBC model, the exact approach used in the cytoskeletal dynamics model would not work. Instead, the LJ force-extension relationship is integrated into the WLC-POW force-extension relationship, forming the WLC-POW-LJ force-extension relationship. The LJ force-extension relationship was selected because, in addition to being used in the cytoskeletal dynamics model, the force response goes to zero for large extensions, thereby allow for the dissociations to occur. In addition, the link automatically reassociates when the two actin nodes come within a threshold distance of one another.

The LJ is commonly used in atomistic MD simulations, since the term raised to the 6th power directly relates to the van der Waals forces. The scale of the hRBC model is much greater, so the LJ may not accurately account for the behavior of the spectrin chains at large extensions. Other force-extension

relationships that exhibit the same qualitative behavior as the LJ may actually provide more accurate behavior when incorporated into the WLC-POW force-extension expression.

The other expressions for interaction potentials (the area, volume, and bending) are simple harmonic approximations. They may be adequate for small deformations of the hRBC, but not for the large deformation the hRBC experiences every cycle through the body. A more accurate term for the area conservation could be explicitly based on the microscopic “conservation” of area of a lipid bilayer due to the hydrophobic nature of the lipid tails. A more accurate term for the volume conservation could be explicitly based on the ion exchange across the cell membrane (and the resulting osmosis) [129]. In addition, the parameters for the area and volume conservation were chosen rather arbitrarily with the only constraint being that their magnitude had to be large enough to ensure the area “incompressibility” and volume incompressibility of the membrane and the hRBC as a whole, respectively. More accurate physical parameters would result in a more accurate model. The derivation of the bending interaction potential in [44] assumed that the hRBC was a sphere. A more accurate term for the bending interaction potential could take the actual biconcave disc morphology of the hRBC into account. This formulation of the bending term may result in a bending interaction potential that varies with location on the hRBC surface, which may help to ensure that the hRBC model exhibits the observed shape memory behavior [125].

The Combined hRBC model is unable to reproduce the vast morphologies of the cell that are observed in experiments [19]. However, mathematical expressions for a subset of hRBC shapes exist [130]. These shapes include the stomatocyte and the echinocyte. The use of these expressions in the derivation of the hRBC model may allow for simulations to produce hRBCs with these shapes.

Incorporating a more biologically accurate model for the spectrin-actin junction complex will improve simulation accuracy. The current model treats the actin node as a point mass, when in fact it is a short, semi-cylindrical protofilament. In addition, the spectrin links attach at multiple points along the length of the protofilament, resulting in the generation of moments and the rotation of the protofilament, which in turn could result in a non-spherically symmetric interaction between topologically connected nodes [131, 132, 133, 134, 135]. A revision of the model for the spectrin-actin junction complex may also have an effect on the chemical reactions used to model the dissociation and reassociation of the spectrin links from the actin nodes. A simple, two-state model is currently implemented in the metabolism. The structure of the junction complex contains more than just two components (see [FIGURE 2-4](#)). There is evidence that a different protein (protein 4.1) is actually the subject of the phosphorylation/dephosphorylation reactions that are responsible for the dissociation/reassociation of the spectrin links [136, 137]. Better equations in the metabolic model for the dissociation/reassociation reactions would help significantly in the reproduction of the actual hRBC behaviors [138, 139].



The method of transforming the number of links to a concentration is rather ad hoc. The total volume of the hRBC is used, when the cytoskeleton actually only resides on the inner surface of the membrane. This transformation method was employed out of necessity, and its form was chosen purely for its simplicity. A more accurate method could employ an effective volume that better represents the actual volume occupied by the cytoskeleton of the hRBC.

Currently, the reaction rates for the dissociation and reassociation of the spectrin links in the Combined hRBC model are constant. Reaction rates for proteins (and in general), however, are functions of external forces acting on the participants [31]. A more accurate model could use reaction rates that are functions of the force and/or extension of a specific link [140, 141, 142].

The modified reaction rates could be a function of the average extension of the spectrin links in the model and/or a function of the average force exerted by the spectrin links. In this case, all the spectrin link would be subject to the same reaction rate. Alternatively, the reaction rate for a specific spectrin link could be a function of the extension of that link and/or the force exerted by that link alone. In this case, each spectrin link would potentially experience a different reaction rate. The first case would be simpler to implement, since the MMwM model assumes a continuum nature to the reactions. The second case would be much more difficult to implement. It may be impossible to incorporate dissociation-reassociation reactions of the spectrin links into the metabolic model, as was done in the Combined hRBC model.

Currently, the viscous effects due to the hRBC membrane, the cytoplasm, and the blood plasma are accounted for in the Combined hRBC model through the use of [EQUATION 3.29](#) and [EQUATION 3.30](#). While these equations represent an improvement over the previous models for viscous effects, there is still room for improvement in the model. Several attempts have been made by Lubarda et al. in an effort to account for the viscoelasticity of the hRBC through the use of a continuum model of a thin membrane [143, 144, 145]. This continuum approach is relevant to the modeling of the hRBC because a more complete viscoelastic model known as the standard linear solid (SLS) model is used. The SLS model is able to reproduce stress relaxation behavior better than the Kelvin-Voigt model of viscoelasticity that is currently used in the Combined hRBC model. A separate effort by Kloppel et al. uses a discrete, finite element approach to account for the nonlinear elastic and viscoelastic behaviors of the hRBC [146]. This discrete approach could be used to incorporate the continuum equations of viscoelasticity used by Lubarda et al. into the Combined hRBC model.

### 5.2.2 Comprehensiveness of Model

The primary objective of this thesis was to combine current models of the hRBC in order to create a more “complete” model that is able to reproduce of the observed hRBC behaviors. Despite the contributions from this thesis, the Combined hRBC model is still unable to reproduce many of the observed hRBC behaviors. A number of modifications to the Combined hRBC model would improve the “completeness” of this model.

Enhancements to the metabolic model used in the Combined hRBC model would provide a huge step towards being able to accurately reproduce the hRBC metabolism in simulations of the Combined hRBC model. The MMwM model [15] has 56 metabolites and 53 reactions, but these are only a fraction of the metabolites and reactions found in the actual hRBC. The true number is currently unknown, but other metabolic models with more metabolites and reactions exist. The iAB-RBC-283 model of the hRBC metabolism [147], for example, has 292 intracellular reactions, 77 transporters, and 267 unique small metabolites. Unfortunately, the iAB-RBC-283 model does not include any information on the kinetics of the intracellular reactions, which means that only steady-state analysis of the hRBC metabolism can be performed using the iAB-RBC-283 model. The addition of rate equations to the iAB-RBC-283 model, however, would make it a more complete model of hRBC metabolism. The form of the rate equations could be determined through the use of available biochemical resources [148], and the parameters in the rate equations could be determined through additional experiments on the hRBC metabolism.

The main function of the hRBC is the delivery of oxygen to tissues, but the Combined hRBC model does not include any provisions to account for that oxygen transport. A more complete model should include the oxygenation and deoxygenation of the hemoglobin in the hRBC as part of the circulatory cycle.

Some simulations of the CG-hRBC model traveling through an artificial heart valve were performed by Hussein Ezzeldin as part of his thesis [1]. In these simulations, however, the hRBC model was treated as a point mass traveling along pathline obtained from independent computational fluid dynamics (CFD) simulations of the heart valves [149]. This treatment of the hRBC model prevents some of the more interesting dynamic behaviors of the hRBC (such as the tumbling to tank-treading transition [150]) from being observed in simulations. Simulations of a single hRBC model in blood flow [151] would provide some insight into these behaviors. These simulations could be elaborated upon to better include the actual characteristics of blood flow in the human body, such as its pulsatile nature. Full simulations of multiple discrete hRBC models in blood flow have yet to be performed. It still may be a number of years before these simulations can be performed due to computational limitations, but they would provide the most accurate and complete representation of blood flow in the human body.

As mentioned in [SECTION 5.1](#), one of the initial goals of this thesis was to include the mechanosensitivity of the actual hRBC in the Combined hRBC model. Unfortunately, at the time of this thesis, not enough is known about the mechanisms behind the observed mechanosensitivity for it to be incorporated in

the model at this time [152, 153]. The leading hypothesis is that the calcium concentration in the hRBC serves as a signaling mechanism [154, 155, 156, 157]. It would be very interesting to see the effect that this mechanosensitivity would have on simulations of the Combined hRBC model, especially simulations of the model in under the conditions experienced in blood flow.

### 5.2.3 Computational Improvements

A considerable amount of effort was expended to increase the speed of the hRBC simulations. Initial versions of the code were too slow to allow the simulation to run to a point where conclusion could be drawn about the results. That being said, there are still multiple areas of the simulation where significant improvements in simulation speed can be gained.

- Restructuring code architecture to run in parallel to improve computation speed.
- Conversion of code to another language (e.g. C, C++, Fortran) that would provide better computation speed.
- Usage of a better numerical integration algorithm to obtain the solution of the equations of motion. The Velocity Verlet algorithm was used for numerical integration of the equations of motion because it offers greater accuracy than the Euler method without any additional computational cost. Other numerical integration methods were not investigated in this thesis. It is possible that another method, although more computationally expensive, may allow for the use of a longer time step length, thereby resulting in faster simulations.
- Usage of a variable time step that changes dynamically during simulation to decrease the total number of time steps required to complete a simulation
- Conversion of the hRBC model to an even more coarse-grained version to decrease the number of degrees of freedom in the system and increase

simulation speed. Even when it was highly coarse-grained ( $N_v = 250$ ), the CG-hRBC model was still able to adequately reproduce an accurate and smooth hRBC shape description with correct mechanical deformation results [44]. The Combined hRBC model corresponds to the spectrin-level model of the CG-hRBC (the number of degrees of freedom are not reduced through the systematic coarse-graining procedure used by Fedosov et al.). In the spectrin-level model, each link in the model directly represents a spectrin link in the cytoskeleton of the hRBC. This high level of detail was required because the links in the model are individually dissociated/reassociate as dictated by the mechanical-metabolic connection described earlier. If the hRBC model was systematically coarse-grained, additional provisions would need to be introduced in order to correct the mechanical-metabolic connection, since a link in the coarse-grained model no longer corresponds to a spectrin link in the actual hRBC.

- Increasing the robustness of the code. Currently, the evaluation of certain mathematical expressions in the code with particular values of simulation variables (e.g. zero, Inf, NaN) will result in errors that prevent the simulations from converging. There are some error checks built into the code that check the values of specific variables each time step in order to determine if the values of those variable will cause an error when they are used by the simulation. If an error check is tripped, the simulation is halted and the simulation data is saved. This approach helps mitigate

some headaches that result from performing simulation, but it is in no way a complete solution to the problem. In addition, the data that is saved is data from a simulation that is unstable and it may be unusable. The error checks themselves only check the values of several variables. There are other conditions that may cause a simulation to become unstable that are not accounted for by the currently implemented error checks. There are also conditions that will not trip an error check, but still results in simulation data that is unusable. There are also no provisions in the code to prevent invalid simulation parameters from being used, specifically too large of a time step. Ideally, the simulation would be able to predict far enough ahead of time that the simulation is becoming unstable and make adjustments in order to keep the simulation stable. The simulation would also be able to recognize when invalid simulation parameters are used.



# Appendix

## A.1 Code Files

This appendix contains the code files used as part of the simulation environment. The files are provided in alphabetical order. The code files are distributed in the hope that they will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

The code files associated with the *distmeshsurface.m* set of functions are the work of Professor Per-Olof Persson and Professor Gilbert Strang [81]. These code files are protected under the GNU GPL and are not included in this appendix to avoid license and copyright infringement. At the time of this thesis, all the associated code files are available on Professor Per-Olof Persson's web page: <http://persson.berkeley.edu/distmesh/>.

Additionally, the code files associated with the *export\_fig.m* set of functions (that were used to produce many of the plots in this thesis) are the work of Oliver Woodford [158] and are not included in this this appendix to avoid license and copyright infringement. At the time of this thesis, all the associated code files are available on the MATLAB® Central File Exchange: <http://www.mathworks.com/matlabcentral/fileexchange/23629-exportfig>

### A.1.1 f\_connectivity.m

```
function [TRILinks,bending_pts] = f_connectivity(Links,Elements) %#codegen
%
% Generates TRILinks and bending_pts arrays using Links and Elements arrays as
% inputs
%
% Written by Hussein Ezzeldin
% Reproduced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hrBC_Triangulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   Links: array of nodes that form links
%   Elements: array of nodes that form elements
%
% OUTPUTS:
%   TRILinks: array of neighboring elements
%   bending_pts: array of nodes that form bending points
%
% GLOBAL VARIABLES:
%   none
%
% SAVE FILES:
%   none

k = 1;
nele = length(Elements);
ns = 3*nele/2;
TRILinks = zeros(length(Links),2);
bending_pts = zeros(length(Links),4);

    for i = 1:ns

        kk = 1;

        for j = 1:nele

            if Links(i,1) == Elements(j,1) || Links(i,1) == Elements(j,2) ||
Links(i,1) == Elements(j,3)

                if Links(i,2) == Elements(j,1) || Links(i,2) == Elements(j,2) ||
Links(i,2) == Elements(j,3)

                    TRILinks(k,kk) = j;
```

```

        if kk == 1

            for ii = 1:3

                if Elements(j,ii) ~= Links(i,1) && Elements(j,ii) ~=
Links(i,2)

                    bending_pts(i,1:3) = [Elements(j,ii) Links(i,:)];

                end

            end

        elseif kk == 2

            for ii = 1:3

                if Elements(j,ii) ~= Links(i,1) && Elements(j,ii) ~=
Links(i,2)

                    bending_pts(i,4) = Elements(j,ii);

                end

            end

        end

        kk = kk + 1;

        if kk > 2

            break

        end

    end

end

k = k + 1;

end

end

```

## A.1.2 f\_cons\_forces.m

```
function [F_C,U1,U2,U3,LEN,Ak,AREA] =  
f_cons_forces(Links,Elements,bending_pts,r,CON) %#codegen  
%  
% Calculates conservative forces acting on nodes in the Combined hrBC model  
%  
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed  
% Mechanical-Metabolic Model of the Human Red Blood Cell".  
%  
% CALLED BY:  
%   hrBC_Stretching_Simulation.m  
%  
% LOAD FILES:  
%   none  
%  
% INPUTS:  
%   Links: array of nodes that form links  
%   Elements: array of nodes that form elements  
%   bending_pts: array of nodes that form bending points  
%   r: array of node locations  
%   CON: vector of connectivity of links  
%  
% OUTPUTS:  
%   F_C: array of conservative forces acting on nodes  
%   U1: current link potential energy  
%   U2: current bending potential energy  
%   U3: current local area potential energy  
%   LEN: vector of link lengths  
%   Ak: vector of areas of individual elements  
%   AREA: global area  
%  
% GLOBAL VARIABLES:  
%   A0: equilibrium local area  
%   A0t: equilibrium global area  
%   adjx: effective length in LJ  
%   Eng: association energy of spectrin-actin bond  
%   kB: Boltzmann constant  
%   ka: global area constraint  
%   kb: bending constraint  
%   kd: local area constraint  
%   kp: POW constraint  
%   kv: volume constraint  
%   Lm: contour length of links  
%   Lp: persistence length of links  
%   m: exponent in POW  
%   Ns: number of links  
%   Nt: number of elements  
%   Nv: number of nodes  
%   sig: characteristic interaction length scale  
%   T: temperature
```

```

% theta0: equilibrium bending angle
% V0t: equilibrium volume
% VOLUME: actual volume
% xcut: location of maximum attractive force in LJ
%
% SAVE FILES:
% none

% Declare Global Variables and Initialize Terms:
global A0 A0t adjx Eng kB ka kb kd kp kv Lm Lp m Ns Nt Nv sig T theta0 V0t VOLUME
xcut

f_links = zeros(Nv,3); % initialize in-plane force array [N]
f_area_loc = zeros(Nv,3); % initialize local area force array [N]
f_area_g = zeros(Nv,3); % initialize global area force array [N]
f_volume = zeros(Nv,3); % initialize volume force array [N]
f_bending = zeros(Nv,3); % initialize bending force array [N]
F_C = zeros(Nv,3); % initialize total force array [N]

LEN = zeros(Ns,1); % initialize link length vector [m]
Ak = zeros(Nt,1); % initialize element-wise area vector [m^2]
Vk = zeros(Nt,1); % initialize element-wise volume vector [m^3]

U1 = 0; % initialize links energy term [J]
U2 = 0; % initialize bending energy term [J]
U3 = 0; % initialize local area energy term [J]

% Calculate In-Plane and Bending Forces and Energies:
for i = 1:1:Ns

    % In-Plane (2-point interaction):
    a1 = Links(i,1); % index of node 1 [unitless]
    b1 = Links(i,2); % index of node 2 [unitless]

    x_a1 = r(a1,1); % x-coordinate of node 1 [m]
    y_a1 = r(a1,2); % y-coordinate of node 1 [m]
    z_a1 = r(a1,3); % z-coordinate of node 1 [m]
    x_b1 = r(b1,1); % x-coordinate of node 2 [m]
    y_b1 = r(b1,2); % y-coordinate of node 2 [m]
    z_b1 = r(b1,3); % z-coordinate of node 2 [m]

    r_21_1 = [(x_b1 - x_a1) (y_b1 - y_a1) (z_b1 - z_a1)]; % relative position
    vector (from node 1 to node 2) [m]
    len = sqrt(r_21_1(1,1)*r_21_1(1,1) + r_21_1(1,2)*r_21_1(1,2) +
    r_21_1(1,3)*r_21_1(1,3)); % distance between node 1 and node 2 [m]
    e_21_1 = r_21_1./len; % unit relative position vector (from node 1 to node
    2) [m]
    LEN(i,1) = len; % save length of current link [m]
    xn = len/Lm; % normalized length of spring extension [unitless]

    f_wlc_1 = (-kB*T/Lp)*(1/(4*(1 - xn)^2) - (1/4) + xn).*(-e_21_1); % WLC
    contribution to force on node 1 [N]
    f_wlc_2 = -f_wlc_1; % WLC contribution to force on node 2 [N]

```

```

f_pow_1 = (kp/(len^m)).*(-e_21_1); % POW contribution to force on node 1
[N]
f_pow_2 = -f_pow_1; % POW contribution to force on node 2 [N]
f_lj_1 = 24*Eng*(2*(sig^12)/((len - adjx)^13) - (sig^6)/((len -
adjx)^7)).*(-e_21_1); % LJ contribution to force on node 1 [N]
f_lj_2 = -f_lj_1; % LJ contribution to force on node 2 [N]

h1 = (xcut - len) >= 0; % logical Heaviside (extension less than cutoff;
in WLC-POW regime) [unitless]
h2 = (len - xcut) >= 0; % logical Heaviside (extension greater than
cutoff; in POW-LJ regime) [unitless]

f_links(a1,:) = f_pow_1 + f_wlc_1*h1*CON(i,1) + f_lj_1*h2*CON(i,1) +
f_links(a1,:); % total in-plane force on node 1 [N]
f_links(b1,:) = f_pow_2 + f_wlc_2*h1*CON(i,1) + f_lj_2*h2*CON(i,1) +
f_links(b1,:); % total in-plane force on node 2 [N]

U_wlc = (kB*T*Lm/(4*Lp))*((3*xn^2 - 2*xn^3)/(1 - xn)); % WLC contribution
to in-plane potential energy [J]
U_pow = kp/((m - 1)*len^(m - 1)); % POW contribution to in-plane potential
energy [J]
U_lj = 4*Eng*((sig/(len - adjx))^12 - (sig/(len - adjx))^6); % LJ
contribution to in-plane potential energy [J]
U1 = U1 + U_wlc*h1*CON(i,1) + U_pow + U_lj*h2*CON(i,1); % total in-plane
potential energy [J]

% Bending (4-point interaction):
a2 = bending_pts(i,1); % index of node 1 [unitless]
b2 = bending_pts(i,2); % index of node 2 [unitless]
c2 = bending_pts(i,3); % index of node 3 [unitless]
d2 = bending_pts(i,4); % index of node 4 [unitless]

x_a2 = r(a2,1); % x-coordinate of node 1 [m]
y_a2 = r(a2,2); % y-coordinate of node 1 [m]
z_a2 = r(a2,3); % z-coordinate of node 1 [m]
x_b2 = r(b2,1); % x-coordinate of node 2 [m]
y_b2 = r(b2,2); % y-coordinate of node 2 [m]
z_b2 = r(b2,3); % z-coordinate of node 2 [m]
x_c2 = r(c2,1); % x-coordinate of node 3 [m]
y_c2 = r(c2,2); % y-coordinate of node 3 [m]
z_c2 = r(c2,3); % z-coordinate of node 3 [m]
x_d2 = r(d2,1); % x-coordinate of node 4 [m]
y_d2 = r(d2,2); % y-coordinate of node 4 [m]
z_d2 = r(d2,3); % z-coordinate of node 4 [m]

r_21_2 = [(x_b2 - x_a2) (y_b2 - y_a2) (z_b2 - z_a2)]; % relative position
vector (from node 1 to node 2) [m]
r_31_2 = [(x_c2 - x_a2) (y_c2 - y_a2) (z_c2 - z_a2)]; % relative position
vector (from node 3 to node 1) [m]
r_34_2 = [(x_c2 - x_d2) (y_c2 - y_d2) (z_c2 - z_d2)]; % relative position
vector (from node 3 to node 4) [m]
r_24_2 = [(x_b2 - x_d2) (y_b2 - y_d2) (z_b2 - z_d2)]; % relative position
vector (from node 2 to node 4) [m]

```

```

r_32_2 = [(x_c2 - x_b2) (y_c2 - y_b2) (z_c2 - z_b2)]; % relative position
vector (from node 3 to node 2) [m]
r_13_2 = -r_31_2; % relative position vector (from node 1 to node 3) [m]
r_42_2 = -r_24_2; % relative position vector (from node 4 to node 2) [m]
r_23_2 = -r_32_2; % relative position vector (from node 2 to node 3) [m]

tc_1 = [(x_a2 + x_b2 + x_c2) (y_a2 + y_b2 + y_c2) (z_a2 + z_b2 +
z_c2)]./3; % center of mass of element 1 [m]
tc_2 = [(x_b2 + x_c2 + x_d2) (y_b2 + y_c2 + y_d2) (z_b2 + z_c2 +
z_d2)]./3; % center of mass of element 2 [m]

ksi_1 = vcross(r_21_2,r_31_2); % normal vector from surface of element 1
zeta_1 = vcross(r_34_2,r_24_2); % normal vector from surface of element 2

ksi_1_n = ksi_1(1,1)*ksi_1(1,1) + ksi_1(1,2)*ksi_1(1,2) +
ksi_1(1,3)*ksi_1(1,3);
zeta_1_n = zeta_1(1,1)*zeta_1(1,1) + zeta_1(1,2)*zeta_1(1,2) +
zeta_1(1,3)*zeta_1(1,3);
ksi_1_ns = sqrt(ksi_1_n);
zeta_1_ns = sqrt(zeta_1_n);

cosine = sum((ksi_1/ksi_1_ns).*(zeta_1/zeta_1_ns));
check = sum((ksi_1 - zeta_1).*(tc_1 - tc_2));

if cosine > 0.9999995000000042

    cosine = 0.9999995000000042;

end

if check >= 0

    sine = sqrt(1 - cosine^2);

else

    sine = -sqrt(1 - cosine^2);

end

beta_bending = kb*(sine*cos(theta0) - cosine*sin(theta0))/sine; % bending
force coefficient

b11 = -beta_bending*cosine/(ksi_1_n);
b12 = beta_bending/(ksi_1_ns*zeta_1_ns);
b22 = -beta_bending*cosine/(zeta_1_n);

f_bending_1 = b11.*vcross(ksi_1,r_32_2) + b12.*vcross(zeta_1,r_32_2); %
bending force 1 [N]
f_bending_2 = b11.*vcross(ksi_1,r_13_2) + b12.*(vcross(ksi_1,r_34_2) +
vcross(zeta_1,r_13_2)) + b22*vcross(zeta_1,r_34_2); % bending force 2 [N]
f_bending_3 = b11.*vcross(ksi_1,r_21_2) + b12.*(vcross(ksi_1,r_42_2) +
vcross(zeta_1,r_21_2)) + b22*vcross(zeta_1,r_42_2); % bending force 3 [N]

```

```

        f_bending_4 = b12.*vcross(ksi_1,r_23_2) + b22.*vcross(zeta_1,r_23_2); %
bending force 4 [N]

        f_bending(a2,:) = f_bending_1 + f_bending(a2,:); % bending force on node 1
[N]
        f_bending(b2,:) = f_bending_2 + f_bending(b2,:); % bending force on node 2
[N]
        f_bending(c2,:) = f_bending_3 + f_bending(c2,:); % bending force on node 3
[N]
        f_bending(d2,:) = f_bending_4 + f_bending(d2,:); % bending force on node 4
[N]

        v_1 = cosine; % sum(ksi_1.*zeta_1)/(ksi_1_ns*zeta_1_ns);

        if v_1 <= - 1

            v_2 = -1;

        elseif v_1 >= 1

            v_2 = 1;

        else

            v_2 = v_1;

        end

        theta = acos(v_2); angle between elements [rad]
        U2 = U2 + kb*(1 - cos(theta - theta0)); % total bending potential energy
[J]

    end

% Calculate Area and Volume:
for i = 1:1:Nt

    a3 = Elements(i,1); % index of node 1 [unitless]
    b3 = Elements(i,2); % index of node 2 [unitless]
    c3 = Elements(i,3); % index of node 3 [unitless]

    x_a3 = r(a3,1); % x-coordinate of node 1 [m]
    y_a3 = r(a3,2); % y-coordinate of node 1 [m]
    z_a3 = r(a3,3); % z-coordinate of node 1 [m]
    x_b3 = r(b3,1); % x-coordinate of node 2 [m]
    y_b3 = r(b3,2); % y-coordinate of node 2 [m]
    z_b3 = r(b3,3); % z-coordinate of node 2 [m]
    x_c3 = r(c3,1); % x-coordinate of node 3 [m]
    y_c3 = r(c3,2); % y-coordinate of node 3 [m]
    z_c3 = r(c3,3); % z-coordinate of node 3 [m]

    r_13_3 = [(x_a3 - x_c3) (y_a3 - y_c3) (z_a3 - z_c3)]; % relative position
vector (from node 1 to node 3) [m]

```



```

        r_21_3 = [(x_b3 - x_a3) (y_b3 - y_a3) (z_b3 - z_a3)]; % relative position
vector (from node 2 to node 1) [m]
        r_31_3 = -r_13_3; % relative position vector (from node 3 to node 1) [m]

        tc_3 = [(x_a3 + x_b3 + x_c3) (y_a3 + y_b3 + y_c3) (z_a3 + z_b3 + z_c3)]/3;
% location of center of mass of individual element [m]
        ksi_2 = vcross(r_21_3,r_31_3); % vector normal to surface of element [m^2]
        Ak(i,1) = sqrt(ksi_2(1,1)*ksi_2(1,1) + ksi_2(1,2)*ksi_2(1,2) +
ksi_2(1,3)*ksi_2(1,3))/2; % area contribution of individual element [m^2]
        vk(i,1) = sum(ksi_2.*tc_3)/6; % volume contribution of individual element
[m^3]

    end

    AREA = sum(Ak); % sum individual element area contributions to find total area
[m^2]
    VOLUME = sum(vk); % sum individual element volume contributions to find total
volume [m^3]
    beta_area = -ka*(AREA - A0t)/A0t; % coefficient in global area force
calculation

    % Calculate Local Area, Global Area and Volume Forces and Local Area Energies:
    for i = 1:1:Nt

        a3 = Elements(i,1); % index of node 1 [unitless]
        b3 = Elements(i,2); % index of node 2 [unitless]
        c3 = Elements(i,3); % index of node 3 [unitless]

        x_a3 = r(a3,1); % x-coordinate of node 1 [m]
        y_a3 = r(a3,2); % y-coordinate of node 1 [m]
        z_a3 = r(a3,3); % z-coordinate of node 1 [m]
        x_b3 = r(b3,1); % x-coordinate of node 2 [m]
        y_b3 = r(b3,2); % y-coordinate of node 2 [m]
        z_b3 = r(b3,3); % z-coordinate of node 2 [m]
        x_c3 = r(c3,1); % x-coordinate of node 3 [m]
        y_c3 = r(c3,2); % y-coordinate of node 3 [m]
        z_c3 = r(c3,3); % z-coordinate of node 3 [m]

        r_32_3 = [(x_c3 - x_b3) (y_c3 - y_b3) (z_c3 - z_b3)]; % relative position
vector (from node 3 to node 2) [m]
        r_13_3 = [(x_a3 - x_c3) (y_a3 - y_c3) (z_a3 - z_c3)]; % relative position
vector (from node 1 to node 3) [m]
        r_21_3 = [(x_b3 - x_a3) (y_b3 - y_a3) (z_b3 - z_a3)]; % relative position
vector (from node 2 to node 1) [m]
        r_31_3 = -r_13_3; % relative position vector (from node 3 to node 1) [m]

        ksi_2 = vcross(r_21_3,r_31_3); % normal vector from surface of element
[m^2]

        % Local Area (3-point interactions):
        alpha_area_loc = -kd*(Ak(i,1) - A0)/(4*A0*Ak(i,1)); % local area force
coefficient

```

```

    f_area_loc_1 = alpha_area_loc.*vcross(ksi_2,r_32_3); % local area force 1
[N]
    f_area_loc_2 = alpha_area_loc.*vcross(ksi_2,r_13_3); % local area force 2
[N]
    f_area_loc_3 = alpha_area_loc.*vcross(ksi_2,r_21_3); % local area force 3
[N]

    f_area_loc(a3,:) = f_area_loc_1 + f_area_loc(a3,:); % local area force on
node 1 [N]
    f_area_loc(b3,:) = f_area_loc_2 + f_area_loc(b3,:); % local area force on
node 2 [N]
    f_area_loc(c3,:) = f_area_loc_3 + f_area_loc(c3,:); % local area force on
node 3 [N]

    U3 = U3 + kd*((Ak(i,1) - A0)^2)/(2*A0); % total local area energy
potential energy [J]

    % Global Area (3-point interactions):
    alpha_area_g = beta_area/(4*Ak(i,1)); % global area force coefficient

    f_area_g_1 = alpha_area_g.*vcross(ksi_2,r_32_3); % global area force 1 [N]
    f_area_g_2 = alpha_area_g.*vcross(ksi_2,r_13_3); % global area force 2 [N]
    f_area_g_3 = alpha_area_g.*vcross(ksi_2,r_21_3); % global area force 3 [N]

    f_area_g(a3,:) = f_area_g_1 + f_area_g(a3,:); % global area force on node
1 [N]
    f_area_g(b3,:) = f_area_g_2 + f_area_g(b3,:); % global area force on node
2 [N]
    f_area_g(c3,:) = f_area_g_3 + f_area_g(c3,:); % global area force on node
3 [N]

    % Volume (3-point interactions):
    beta_volume = -kv*(VOLUME - V0t)/V0t; % volume force coefficient
    bv = beta_volume/6;
    tc_3 = [(x_a3 + x_b3 + x_c3) (y_a3 + y_b3 + y_c3) (z_a3 + z_b3 + z_c3)]/3;
% center of mass of element [m]

    f_volume_1 = bv.*(ksi_2./3 + vcross(tc_3,r_32_3)); % volume force 1 [N]
    f_volume_2 = bv.*(ksi_2./3 + vcross(tc_3,r_13_3)); % volume force 1 [N]
    f_volume_3 = bv.*(ksi_2./3 + vcross(tc_3,r_21_3)); % volume force 1 [N]

    f_volume(a3,:) = f_volume_1 + f_volume(a3,:); % volume force on node 1 [N]
    f_volume(b3,:) = f_volume_2 + f_volume(b3,:); % volume force on node 2 [N]
    f_volume(c3,:) = f_volume_3 + f_volume(c3,:); % volume force on node 3 [N]

end

    F_C = f_links + f_area_g + f_area_loc + f_volume + f_bending; % total
conservative force on all nodes [N]

end

```

### A.1.3 f\_diss.m

```
function [WT,CON,WEIGHTS] = f_diss(WT,CON,WEIGHTS,num_dis,nwt,V_links) %#codegen
%
% Dissociates links selected by datasample
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hrBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   WT: vector of wait time counters for dissociated links
%   CON: vector of connectivity of links
%   WEIGHTS: vector of selection weights of links
%   num_dis: number of links to be dissociated
%   nwt: maximum wait time
%   V_links: vector of links selected by datasample
%
% OUTPUTS:
%   WT: vector of wait time counters for dissociated links
%   CON: vector of connectivity of links
%   WEIGHTS: vector of selection weights of links
%
% GLOBAL VARIABLES:
%   none
%
% SAVE FILES:
%   none

% Dissociate selected links and update wait time counters/weights
for i = 1:1:num_dis

    WT(V_links(i,1),1) = nwt; % set number of time steps in wait time counters
    for selected link
        CON(V_links(i,1),1) = 0; % set connectivity of selected link to 0
        WEIGHTS(V_links(i,1),1) = 0; % set weight of selected link to 0

    end

end

end
```

### A.1.4 f\_diss\_forces.m

```
function F_D = f_diss_forces(Links,vel,F_C) %%codegen
%
% Calculates dissipative forces on nodes in the Combined hRBC model
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hRBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   Links: array of nodes that form links
%   vel: array of node velocities
%   F_C: array of conservative forces acting on nodes
%
% OUTPUTS:
%   F_D: array of dissipative forces acting on nodes
%
% GLOBAL VARIABLES:
%   ma: mass of an actin node
%   Ns: number of links
%   Nv: number of nodes
%   ni: viscosity of cytoplasm
%   nm: viscosity of cell membrane
%   no: viscosity of blood plasma
%   rho_a: density of an actin node
%
% SAVE FILES:
%   none

global ma Ns Nv ni nm no rho_a

F_D_r = zeros(Nv,3); % initialize relative dissipative force array
F_D_a = zeros(Nv,3); % initialize absolute dissipative force array

Cf = sqrt(no*ni); % combined fluid viscosity [N s/m^2]
radius = (3*ma/(4*pi*rho_a))^(1/3); % representative radius of actin protein
[m]
Cm = 4*nm/sqrt(3); % membrane damping coefficient [N s/m] = 0.050806823688687

% Relative Dissipative Force Calculation (Membrane Damping):
for i = 1:1:Ns

    a = Links(i,1); % index of node 1 [unitless]
    b = Links(i,2); % index of node 2 [unitless]
```

```

v_x_a = vel(a,1); % x-velocity of node 1 [m/s]
v_y_a = vel(a,2); % y-velocity of node 1 [m/s]
v_z_a = vel(a,3); % z-velocity of node 1 [m/s]
v_x_b = vel(b,1); % x-velocity of node 2 [m/s]
v_y_b = vel(b,2); % y-velocity of node 2 [m/s]
v_z_b = vel(b,3); % z-velocity of node 2 [m/s]

v_21 = [(v_x_b - v_x_a) (v_y_b - v_y_a) (v_z_b - v_z_a)]; % relative
velocity vector (from node 1 to node 2) [m/s]

F_D_r(a,:) = F_D_r(a,:) + Cm*v_21; % relative dissipative force on node 1
[N]
F_D_r(b,:) = F_D_r(b,:) - Cm*v_21; % relative dissipative force on node 2
[N]

end

% Relative Dissipative Force Magnitude Check:
for i = 1:1:Nv

    n_F_C = norm(F_C(i,:)); % norm of F_C
    n_F_D_r = norm(F_D_r(i,:)); % norm of F_D_r

    if n_F_D_r > n_F_C % if the magnitude of the dissipative force is greater
than the magnitude of the conservative force

        F_D_r(i,:) = F_D_r(i,:)*n_F_C/n_F_D_r; % then, set the dissipative
force to the negative of the conservative force (so that acc = 0 [m/s^2])

    end

end

v_cm_x = mean(vel(:,1)); % x-velocity of center of mass [m/s]
v_cm_y = mean(vel(:,2)); % y-velocity of center of mass [m/s]
v_cm_z = mean(vel(:,3)); % z-velocity of center of mass [m/s]
v_cm = [v_cm_x v_cm_y v_cm_z]; % velocity of center of mass [m/s]

% Absolute Dissipative Force Calculation (Blood Plasma and Cytoplasm Damping):
for i = 1:1:Nv

    F_D_a(i,:) = F_D_a(i,:) - 6*pi*Cf*radius*(v_cm - vel(i,:)); % absolute
dissipative force [N]

end

F_D = F_D_r + F_D_a; % total dissipative force [N]

end

```

## A.1.5 f\_ke.m

```
function KE = f_ke(vel,MASS) %%codegen
%
% Calculates total kinetic energy of Combined hRBC model
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hRBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   vel: vector of wait time counters for dissociated links
%   MASS: nodal mass
%
% OUTPUTS:
%   KE: total kinetic energy
%
% GLOBAL VARIABLES:
%   Nv: number of nodes
%
% SAVE FILES:
%   none

global Nv

    TEMP_1 = zeros(Nv,1);

    for i = 1:1:Nv

        TEMP_1(i,1) = vel(i,1)*vel(i,1) + vel(i,2)*vel(i,2) + vel(i,3)*vel(i,3); %
squared velocity of node [m^2/s^2] squared velocity of node [m^2/s^2]

    end

    KE = 0.5*MASS*sum(TEMP_1); % total kinetic energy [J]

end
```

## A.1.6 f\_metabolism.m

```
function dx = f_metabolism(~,X) %#codegen
%
% Calculates reaction rates of metabolism in Combined hRBC model
%
% Adapted From "Modelling Metabolism with Mathematica"
% By Peter J. Mulquiney and Philip W. Kuchel
%
% Reproduced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hRBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   X: vector of metabolite concentration
%
% OUTPUTS:
%   dx: vector of rates of change of metabolite concentrations
%
% GLOBALS VARIABLE:
%   VOLUME: actual volume
%   CONC_SA_max: maximum possible concentration of associated spectrin
%               links
%
% SAVE FILES:
%   none

global VOLUME CONC_SA_max

dx = zeros(57,1); % initialize rate of change of metabolite concentrations
vector

% Metabolism Parameters
VHK = 2.5e-8;
Kihkmgatp = 0.0010;
Kmhkmgatp = 0.0010;
Kihkglc = 4.7e-5;
Kihkglc6p = 4.7e-5;
Kihkmgadp = 0.0010;
Kmhkmgadp = 0.0010;
Kdihkbpg = 0.0040;
Kdihkglc16p2 = 3.0e-5;
Kdihkglc6p = 1.0e-5;
Kdihkgsh = 0.0030;
VGPI = 2.18e-7;
Kmgpiglc6p = 1.81e-4;
```

```

Kmgpifru6p = 7.1e-5;
kcatfgpi = 1470.0;
kcatrgpi = 1760.0;
VPFK = 1.1e-7;
Kmpfkfrfu6p = 7.5e-5;
Kmpfkmgatp = 6.8e-5;
Kmpfkfrfu16p2 = 5.0e-4;
Kmpfkmgadp = 5.4e-4;
KTpfkatp = 1.0e-4;
KTpfkmg = 0.0040;
KTpfkb23pg = 0.0050;
KRpfkamp = 3.0e-4;
KRpfkphos = 0.03;
KRpfkg1c16p2 = 0.01;
kcatfpfk = 822.0;
kcatrpfk = 36.0;
Kapfk = 8.91250938133746e-8;
npfk = 5.0;
VALD = 3.7e-7;
Kmaldfru16p2 = 7.1e-6;
Kialdfru16p2 = 1.98e-5;
Kmaldgrnp = 3.5e-5;
Kialdgrnp = 1.1e-5;
Kmaldgrap = 1.89e-4;
Kialdb23pg = 0.0015;
kcatfald = 68.0;
kcatrald = 234.0;
VTPI = 1.14e-6;
Kmtpigrap = 4.46e-4;
Kmtpigrnp = 1.624e-4;
kcatftpi = 14560.0;
kcatrtpi = 1280.0;
VGDH = 7.66e-6;
Kmgapdhnad = 4.5e-5;
Kigapdhnad = 4.5e-5;
Kmgapdhphos = 0.00316;
Kigapdhphos = 0.00316;
Kmgapdhgrap = 9.5e-5;
Kidgapdhgrap = 3.1e-5;
Kmgapdhb13pg = 6.71e-7;
Kidgapdhb13pg = 1.0e-6;
VPGK = 2.74e-6;
Kmpgkmgadp = 1.0e-4;
Kipgkmgadp = 8.0e-5;
Kmpgkb13pg = 2.0e-6;
Kipgkb13pg = 1.6e-6;
Kmpgkmgatp = 0.0010;
Kipgkmgatp = 1.86e-4;
Kmpgkp3ga = 0.0011;
Kipgkp3ga = 2.05e-4;
kcatfpgk = 2290.0;
kcatrpgk = 917.0;
VPGM = 4.1e-7;

```



```

Kmpgmp3ga = 1.68e-4;
Kmpgmp2ga = 2.56e-5;
kcatfpgm = 795.0;
kcatrpgm = 714.0;
VENO = 2.2e-7;
Kienop2ga = 1.4e-4;
Kmenop2ga = 1.4e-4;
Kienopep = 1.105e-4;
Kmenopep = 1.105e-4;
Kienomg = 4.6e-5;
Kmenomg = 4.6e-5;
kcatfeno = 190.0;
kcatreno = 50.0;
VPK = 8.7e-8;
KTPkatp = 0.00339;
KRpkpyr = 0.0020;
KRpkpep = 2.25e-4;
KRpkmgatp = 0.0030;
KRpkmgadp = 4.74e-4;
KRf16p2 = 5.0e-6;
KRg16p2 = 1.0e-4;
kcatfpk = 1386.0;
kcatrpk = 3.26;
VLDH = 3.43e-6;
Kmlldhnadh = 8.44e-6;
Kildhnadh = 2.45e-6;
Kmlldhnad = 1.07e-4;
Kildhnad = 5.03e-4;
Kidldhpyr = 1.01e-4;
kcatfldh = 458.0;
kcatrldh = 115.0;
Kmlldhppy = 4.14e-4;
Kmlldhplac = 4.14e-4;
kfldhp = 0.00346;
krldhp = 5.43e-7;
VG6PDH = 9.3e-8;
kg6pdh1 = 1.1e8;
kg6pdh2 = 870.0;
kg6pdh3 = 2.6e7;
kg6pdh4 = 300.0;
kg6pdh5 = 750.0;
kg6pdh6 = 2000.0;
kg6pdh7 = 220000.0;
kg6pdh8 = 1.1e9;
kg6pdh9 = 10000.0;
kg6pdh10 = 1.4e9;
VLactonase = 1.4e-5;
klactonase1 = 1.3e7;
klactonase2 = 1000.0;
klactonase3 = 29.0;
kspontaneouspglhydrolysis = 7.1e-4;
VP6GDH = 2.1e-6;
kp6gdh1 = 2400000.0;

```

```
kp6gdh2 = 410.0;
kp6gdh3 = 2.0e9;
kp6gdh4 = 26000.0;
kp6gdh5 = 48.0;
kp6gdh6 = 30.0;
kp6gdh7 = 630.0;
kp6gdh8 = 36000.0;
kp6gdh9 = 800.0;
kp6gdh10 = 225000.0;
kp6gdh11 = 300.0;
kp6gdh12 = 4950000.0;
VGSSGR = 1.25e-7;
kgssgr1 = 8.5e7;
kgssgr2 = 510.0;
kgssgr3 = 1.0e9;
kgssgr4 = 72000.0;
kgssgr5 = 810.0;
kgssgr6 = 1000.0;
kgssgr7 = 1000000.0;
kgssgr8 = 5.0e7;
kgssgr9 = 1000000.0;
kgssgr10 = 5.0e7;
kgssgr11 = 7000.0;
kgssgr12 = 1.0e8;
VRu5PE = 4.22e-6;
kru5pe1 = 3910000.0;
kru5pe2 = 438.0;
kru5pe3 = 305.0;
kru5pe4 = 1490000.0;
VR5PI = 1.42e-5;
kr5pi1 = 60900.0;
kr5pi2 = 33.3;
kr5pi3 = 14.2;
kr5pi4 = 21600.0;
ktk1 = 216000.0;
ktk2 = 38.0;
ktk3 = 34.0;
ktk4 = 156000.0;
ktk5 = 329000.0;
ktk6 = 175.0;
ktk7 = 40.0;
ktk8 = 44800.0;
ktk9 = 2240000.0;
ktk10 = 175.0;
ktk11 = 40.0;
ktk12 = 21300.0;
VTA = 6.9e-7;
kta1 = 580000.0;
kta2 = 45.3;
kta3 = 16.3;
kta4 = 1010000.0;
kta5 = 490000.0;
kta6 = 60.0;
```

```

kta7 = 17.0;
kta8 = 79000.0;
kbpbsp2 = 400.0;
kbpbsp5 = 1.0e8;
kbpbsp7 = 1000.0;
kbpbsp8 = 10000.0;
kbpbsp9 = 0.55;
kbpbsp10 = 1979.0;
kbpbsp11 = 0.01;
kbpbsp12 = 1000.0;
kbpbsp14 = 1.0e9;
kbpbsp15 = 610000.0;
kbpbsp16 = 0.19;
khamp = 3090000.0;
kkamp = 1.8;
katpase = 5.85e-4;
kox = 3.4e-5;
koxNADH = 0.0163;
kpyrtransporti = 0.018;
klacttransporti = 0.0036;
kphostransporti = 5.6e-4;
kmgatp = 43200.0;
khatp = 9070000.0;
kmghatp = 748.0;
kkatp = 14.0;
kmgatpd = 1200.0;
kmgadp = 3290.0;
khadp = 5420000.0;
kmghadp = 107.0;
kkadp = 4.8;
kmgadpd = 1200.0;
kmgbpg = 7410.0;
khbpg = 1.62e8;
kmghbpg = 513.0;
kh2bpg = 4270000.0;
kkbpg = 85.1;
kkhbpg = 8.9;
kmgbpgd = 1200.0;
kmgb13pgd = 1200.0;
kmgfru16p2 = 363.0;
khf = 7560000.0;
kmghf = 89.0;
kh2f = 1120000.0;
kkf = 10.7;
kkhf = 3.3;
kmgf16p2d = 1200.0;
khphos = 5680000.0;
kkphos = 3.0;
kmgphosd = 1200.0;
kahb = 2.511886431509582e-7;
khbm gatpd = 1200.0;
khbatpd = 1200.0;
khbadpd = 1200.0;

```

```

khbbpgd = 1200.0;
khbb13pgd = 1200.0;

MgATP_0 = 0.001518299902639;
ksaf = 1e7;
ksar = ksaf/(5*MgATP_0);

alpha = 0.7;
Ht = 0.5;
Ve = 2*VOLUME*(1e3)*Ht;
Vi = alpha*VOLUME*(1e3);
rtv = 0.69;
k1 = 0.15;
pH1 = 7.2;

Glc = 5e-3; % actual glucose concentration
CO2 = 1.2e-3;
Lace = 1.82e-3;
Pyre = 8.5e-5;
Phose = 1.92e-3;

% Reaction Rates (54 Reactions)
vhk = (1/vi)*VHK*Vi*(-1.92792*x(17)*x(30)/((1 + 1.0471285480509*10^7/10^pH1 +
2.81838293126445*10^(-10)*10^pH1)*kihkg1c6p*kmhkmgadp) + 299.16*Glc*x(31)/((1 +
1.0471285480509*10^7/10^pH1 + 2.81838293126445*10^(-
10)*10^pH1)*kihkg1c*kmhkmgatp))/((1 + Glc/Kihkg1c + Glc*x(5)/(Kdihkbpg*kihkg1c) +
Glc*x(16)/(Kdihkg1c16p2*kihkg1c) + Glc*x(17)/(Kdihkg1c6p*kihkg1c) +
x(17)/Kihkg1c6p + Glc*x(20)/(Kdihkgsh*kihkg1c) + x(30)/Kihkmgadp +
x(17)*x(30)/(Kihkg1c6p*kmhkmgadp) + x(31)/Kihkmgatp +
Glc*x(31)/(Kihkg1c*kmhkmgatp)));
vgpi = VGPI*(-x(15)*kcatrgpi/Kmgpifru6p + x(17)*kcatfgpi/Kmgpig1c6p)/((1 +
x(15)/Kmgpifru6p + x(17)/Kmgpig1c6p));
vpfk = VPFK*(-x(14)*kcatfpfk*x(30)/(Kmpfkfru16p2*Kmpfkmgadp) +
x(15)*kcatfpfk*x(31)/(Kmpfkfru6p*Kmpfkmgatp))/((1 + x(14)/Kmpfkfru16p2 +
x(15)/Kmpfkfru6p + x(30)/Kmpfkmgadp + x(14)*x(30)/(Kmpfkfru16p2*Kmpfkmgadp) +
x(31)/Kmpfkmgatp + x(15)*x(31)/(Kmpfkfru6p*Kmpfkmgatp))*(1 +
(1/(10^pH1*kapfk))^npfk*(1 + x(3)/Ktpfkatp)^4*(1 + x(5)/Ktpfk23pg)^4*(1 +
x(29)/Ktpfkmg)^4/((1 + x(14)/Kmpfkfru16p2 + x(15)/Kmpfkfru6p)^4*(1 +
x(2)/KRpfkamp)^4*(1 + x(16)/KRpfg1c16p2)^4*(1 + x(46)/KRpfkphos)^4));
vald = VALD*(x(14)*kcatfald/kmaldf16p2 -
x(18)*x(19)*kcatrald/(Kialdgrnp*kmaldgrap))/((1 + x(19)/Kialdgrnp +
x(14)/kmaldf16p2 + x(18)*x(19)/(Kialdgrnp*kmaldgrap) +
x(14)*x(18)*kmaldgrnp/(Kialdf16p2*Kialdgrnp*kmaldgrap) + (x(5) +
x(33))/Kialdb23pg + x(18)*kmaldgrnp*(1 + (x(5) +
x(33))/Kialdb23pg)/(Kialdgrnp*kmaldgrap));
vtpi = VTPI*(x(18)*kcatftpi/kmtpigrap - x(19)*kcatrtpi/kmtpigrnp)/((1 +
x(18)/kmtpigrap + x(19)/kmtpigrnp);
vgapdh = VGDH*((-5.40304497429925)*10^35*x(4)*x(38)/10^pH1 +
1.45911949685535*10^21*x(18)*x(37)*x(46)/(Kigapdhphos*kmgapdhnad))/((2.195177633774
12*10^20*(1 + 3.16227766016838*10^7/10^pH1 + 1*10^(-10)*10^pH1)*x(4)*(1 +
x(18)/Kidgapdhgrap) + 2.0985345932935*10^18*(1 + 3.16227766016838*10^7/10^pH1 +
1*10^(-10)*10^pH1)*x(18)*(1 + x(18)/Kidgapdhgrap) +
1/Kigapdhnad*2.19517763377412*10^20*(1 + 3.16227766016838*10^7/10^pH1 + 1*10^(-

```

$$\begin{aligned}
& 10) * 10^{\text{pH1}} * X(4) * X(37) + 1/\text{Kigapdhnad} * 2.0985345932935 * 10^{18} * (1 + \\
& 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * X(18) * X(37) + \\
& 1.05427942060955 * 10^{33} * (1 + 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * \text{Kmgapdhb13pg} * X(38) / 10^{\text{pH1}} + 1.05427942060955 * 10^{33} * (1 + \\
& 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * X(4) * X(38) / 10^{\text{pH1}} + \\
& 3.32595319105503 * 10^{30} * (1 + 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * X(18) * X(38) / 10^{\text{pH1}} + 1/\text{Kigapdhphos} * 2.0985345932935 * 10^{18} * (1 + \\
& 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * X(18) * (1 + \\
& X(18) / \text{Kigapdhgrap} * X(46) + 2.0985345932935 * 10^{18} * (1 + \\
& 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * \text{Kmgapdhgrap} * X(37) * X(46) / (\text{Kigapdhphos} * \text{Kmgapdhnad}) + \\
& 2.0985345932935 * 10^{18} * (1 + 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * \text{Kmgapdhgrap} * X(4) * X(37) * X(46) / (\text{Kigapdhb13pg} * \text{Kigapdhphos} * \text{Kmgapdhnad}) + \\
& 2.0985345932935 * 10^{18} * (1 + 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * X(18) * X(37) * X(46) / (\text{Kigapdhphos} * \text{Kmgapdhnad}) + \\
& 1/\text{Kigapdhphos} * 1.05427942060955 * 10^{33} * (1 + 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * \text{Kmgapdhb13pg} * X(38) * X(46) / 10^{\text{pH1}} + 1.05427942060955 * 10^{33} * (1 + \\
& 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * \text{Kmgapdhb13pg} * X(4) * X(38) * X(46) / 10^{\text{pH1}} / (\text{Kigapdhb13pg} * \text{Kigapdhphos}) + \\
& 1/\text{Kigapdhphos} * 3.32595319105503 * 10^{30} * (1 + 3.16227766016838 * 10^7 / 10^{\text{pH1}} + 1 * 10^{(-10)} * 10^{\text{pH1}} * X(18) * X(38) * X(46) / 10^{\text{pH1}}); \\
& \text{vpgk} = \text{VPGK} * (X(4) * \text{kcatfpgk} * X(30) / (\text{Kipgkmgadp} * \text{Kmpgkb13pg}) - \\
& \text{kcatrpgk} * X(31) * X(42) / (\text{Kipgkmgatp} * \text{Kmpgkp3ga})) / (1 + X(4) / \text{Kipgkb13pg} + \\
& X(30) / \text{Kipgkmgadp} + X(4) * X(30) / (\text{Kipgkmgadp} * \text{Kmpgkb13pg}) + X(31) / \text{Kipgkmgatp} + \\
& X(42) / \text{Kipgkp3ga} + X(31) * X(42) / (\text{Kipgkmgatp} * \text{Kmpgkp3ga})); \\
& \text{vpgm} = \text{VPGM} * (-\text{kcatrpgm} * X(41) / \text{Kmpgmp2ga} + \text{kcatfpgm} * X(42) / \text{Kmpgmp3ga}) / (1 + \\
& X(41) / \text{Kmpgmp2ga} + X(42) / \text{Kmpgmp3ga}); \\
& \text{veno} = \text{VENO} * (\text{kcatfeno} * X(29) * X(41) / (\text{Kienomg} * \text{Kmenop2ga}) - \\
& \text{kcatreno} * X(29) * X(45) / (\text{Kienopep} * \text{Kmenomg})) / (1 + X(29) / \text{Kienomg} + X(41) / \text{Kienop2ga} + \\
& X(29) * X(41) / (\text{Kienomg} * \text{Kmenop2ga}) + X(45) / \text{Kienopep} + \\
& X(29) * X(45) / (\text{Kienopep} * \text{Kmenomg})); \\
& \text{vpk} = \text{VPK} * (\text{kcatfvpk} * X(30) * X(45) / (\text{KRpkmgadp} * \text{KRpkpep}) - \\
& \text{kcatrpvk} * X(31) * X(47) / (\text{KRpkmgatp} * \text{KRpkpyr})) / ((1 + X(30) / \text{KRpkmgadp} + X(31) / \text{KRpkmgatp} + \\
& X(45) / \text{KRpkpep} + X(30) * X(45) / (\text{KRpkmgadp} * \text{KRpkpep}) + X(47) / \text{KRpkpyr} + \\
& X(31) * X(47) / (\text{KRpkmgatp} * \text{KRpkpyr})) * (1 + 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}} * (1 + \\
& X(3) / \text{Ktpkatp})^4 / ((1 + X(14) / \text{KRf16p2} + X(16) / \text{KRg16p2})^4 * (1 + X(45) / \text{KRpkpep} + \\
& X(47) / \text{KRpkpyr})^4))); \\
& \text{vldh} = \text{VLDH} * (-1306.64221547056 * \text{kcatrlhdh} * X(28) * X(37) / ((1 + \\
& 6.30957344480193 * 10^6 / 10^{\text{pH1}}) * \text{Kildhnad}) + \\
& 2.5634207529267e4 * \text{kcatflhdh} * X(38) * X(47) / ((1 + 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}}) * \text{Kildhndh})) / (X(37) / \text{Kildhnad} + 1306.64221547056 * X(28) * X(37) / ((1 + \\
& 6.30957344480193 * 10^6 / 10^{\text{pH1}}) * \text{Kildhnad}) + X(38) / \text{Kildhndh} + \\
& 1306.64221547056 * \text{Kmlhdhndh} * X(28) * X(38) / ((1 + \\
& 6.30957344480193 * 10^6 / 10^{\text{pH1}}) * \text{Kildhnad} * \text{Kildhndh}) + \\
& 2.5634207529267e4 * \text{Kmlhdhndh} * X(37) * X(47) / ((1 + 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}}) * \text{Kildhnad} * \text{Kildhndh}) + 2.01262239795994 * 10^7 * X(28) * X(37) * X(47) / ((1 + \\
& 6.30957344480193 * 10^6 / 10^{\text{pH1}}) * (1 + 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}}) * \text{Kildhnad}) + \\
& 2.5634207529267e4 * X(38) * X(47) / ((1 + 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}}) * \text{Kildhndh}) + \\
& 4.88940918937581 * 10^6 * X(28) * X(38) * X(47) / ((1 + 6.30957344480193 * 10^6 / 10^{\text{pH1}}) * (1 + \\
& 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}}) * \text{Kildhndh}) + (1 + X(47) / \text{Kildhpyr}) * (1 + \\
& 1306.64221547056 * \text{Kmlhdhndh} * X(28) / ((1 + 6.30957344480193 * 10^6 / 10^{\text{pH1}}) * \text{Kildhnad}) + \\
& 2.5634207529267e4 * \text{Kmlhdhndh} * X(47) / ((1 + 1.58489319246111 * 10^{(-7)} * 10^{\text{pH1}}) * \text{Kildhndh})));
\end{aligned}$$

```

vldhp = (-krldhp*x(28)*x(39)/kmlldhplac + kfldhp*x(40)*x(47)/kmlldhppy)/(1 +
x(28)/kmlldhplac + x(47)/kmlldhppy);
vbpbsp1 = 2.80950914520767*10^8*x(4)*x(6)/(1 + 1.58489319246111*10^(-
7)*10^pH1) - kbpbsp2*x(11);
vbpbsp2 = 17.4099179927889*x(11)/(1 + 4.78630092322638*10^28/10^(4*pH1));
vbpbsp3 = (-kbpbsp5)*x(9) + 3.25336851380399*10^8*x(7)*x(42)/(1 +
4.78630092322638*10^28/10^(4*pH1));
vbpbsp4 = (-kbpbsp7)*x(8) + 1758.57757502918*x(7)*x(41)/(1 +
4.78630092322638*10^28/10^(4*pH1));
vbpbsp5 = kbpbsp8*x(9) - kbpbsp9*x(12);
vbpbsp6 = kbpbsp10*x(8) - kbpbsp11*x(12);
vbpbsp7 = -6.32139557671725*10^6*x(5)*x(6)/(1 + 1.58489319246111*10^(-
7)*10^pH1) + kbpbsp12*x(12);
vbpbsp8 = kbpbsp14*x(7)*x(46) - kbpbsp15*x(10);
vbpbsp9 = kbpbsp16*x(10);
vg6pdh = VG6PDH*(X(17)*kg6pdh1*kg6pdh3*kg6pdh5*kg6pdh7*kg6pdh9*x(39) -
kg6pdh2*kg6pdh4*kg6pdh6*kg6pdh8*kg6pdh10*x(40)*x(44))/(X(17)*kg6pdh3*kg6pdh5*kg6pd
h7*kg6pdh9 + kg6pdh2*(kg6pdh4*kg6pdh6 + kg6pdh4*kg6pdh7 + kg6pdh5*kg6pdh7)*kg6pdh9
+ kg6pdh1*(kg6pdh4*kg6pdh6 + kg6pdh4*kg6pdh7 + kg6pdh5*kg6pdh7)*kg6pdh9*x(39) +
X(17)*kg6pdh1*kg6pdh3*(kg6pdh5*kg6pdh7 + kg6pdh5*kg6pdh9 + kg6pdh6*kg6pdh9 +
kg6pdh7*kg6pdh9)*x(39) + X(17)*kg6pdh3*kg6pdh5*kg6pdh7*kg6pdh10*x(40) +
kg6pdh2*(kg6pdh4*kg6pdh6 + kg6pdh5*kg6pdh6 + kg6pdh5*kg6pdh7)*kg6pdh10*x(40) +
kg6pdh2*kg6pdh4*kg6pdh6*kg6pdh8*x(44) +
kg6pdh1*kg6pdh4*kg6pdh6*kg6pdh8*x(39)*x(44) + X(17)*kg6pdh1*kg6pdh3*(kg6pdh5 +
kg6pdh6)*kg6pdh8*x(39)*x(44) + X(17)*kg6pdh3*(kg6pdh5 +
kg6pdh6)*kg6pdh8*kg6pdh10*x(40)*x(44) + (kg6pdh2*kg6pdh4 + kg6pdh2*kg6pdh5 +
kg6pdh2*kg6pdh6 + kg6pdh4*kg6pdh6)*kg6pdh8*kg6pdh10*x(40)*x(44));
vpghydrolysis = kspontaneouspghydrolysis*x(44) +
vLactonase*kLactonase3*x(44)/((kLactonase2 + kLactonase3)/kLactonase1 + X(44));
vp6gdh = VP6GDH*(kp6gdh1*kp6gdh3*kp6gdh5*kp6gdh7*kp6gdh9*kp6gdh11*x(39)*x(43)
-
CO2*kp6gdh2*kp6gdh4*kp6gdh6*kp6gdh8*kp6gdh10*kp6gdh12*x(40)*x(49))/(CO2*kp6gdh2*kp
6gdh4*kp6gdh6*kp6gdh8*kp6gdh11 + kp6gdh2*(kp6gdh4*kp6gdh6 + kp6gdh4*kp6gdh7 +
kp6gdh5*kp6gdh7)*kp6gdh9*kp6gdh11 +
CO2*kp6gdh1*kp6gdh4*kp6gdh6*kp6gdh8*kp6gdh11*x(39) + kp6gdh1*(kp6gdh4*kp6gdh6 +
kp6gdh4*kp6gdh7 + kp6gdh5*kp6gdh7)*kp6gdh9*kp6gdh11*x(39) +
CO2*kp6gdh2*kp6gdh4*kp6gdh6*kp6gdh8*kp6gdh12*x(40) + kp6gdh2*(kp6gdh4*kp6gdh6 +
kp6gdh4*kp6gdh7 + kp6gdh5*kp6gdh7)*kp6gdh9*kp6gdh12*x(40) +
kp6gdh3*kp6gdh5*kp6gdh7*kp6gdh9*kp6gdh11*x(43) + CO2*kp6gdh1*kp6gdh3*(kp6gdh5 +
kp6gdh6)*kp6gdh8*kp6gdh11*x(39)*x(43) + kp6gdh1*kp6gdh3*(kp6gdh5*kp6gdh7*kp6gdh9 +
kp6gdh5*kp6gdh7*kp6gdh11 + kp6gdh5*kp6gdh9*kp6gdh11 + kp6gdh6*kp6gdh9*kp6gdh11 +
kp6gdh7*kp6gdh9*kp6gdh11)*x(39)*x(43) +
kp6gdh3*kp6gdh5*kp6gdh7*kp6gdh9*kp6gdh12*x(40)*x(43) +
CO2*kp6gdh2*kp6gdh4*kp6gdh6*kp6gdh8*kp6gdh10*x(49) +
CO2*kp6gdh1*kp6gdh4*kp6gdh6*kp6gdh8*kp6gdh10*x(39)*x(49) +
kp6gdh2*(kp6gdh4*kp6gdh6 + kp6gdh4*kp6gdh7 +
kp6gdh5*kp6gdh7)*kp6gdh10*kp6gdh12*x(40)*x(49) + CO2*(kp6gdh2*kp6gdh4 +
kp6gdh2*kp6gdh5 + kp6gdh2*kp6gdh6 +
kp6gdh4*kp6gdh6)*kp6gdh8*kp6gdh10*kp6gdh12*x(40)*x(49) +
kp6gdh1*kp6gdh3*kp6gdh5*kp6gdh7*kp6gdh10*x(39)*x(43)*x(49) +
CO2*kp6gdh1*kp6gdh3*(kp6gdh5 + kp6gdh6)*kp6gdh8*kp6gdh10*x(39)*x(43)*x(49) +
kp6gdh3*kp6gdh5*kp6gdh7*kp6gdh10*kp6gdh12*x(40)*x(43)*x(49) + CO2*kp6gdh3*(kp6gdh5
+ kp6gdh6)*kp6gdh8*kp6gdh10*kp6gdh12*x(40)*x(43)*x(49));

```

```

vgssgr = VGSSGR*((-
X(20))^2*kgssgr2*kgssgr4*kgssgr6*kgssgr8*kgssgr10*kgssgr12*X(39) +
X(21)*kgssgr1*kgssgr3*kgssgr5*kgssgr7*kgssgr9*kgssgr11*X(40))/(X(20)^2*kgssgr2*kgssgr4*kgssgr6*kgssgr8*kgssgr10 + X(20)*kgssgr2*kgssgr4*kgssgr6*kgssgr8*kgssgr11 +
X(21)*kgssgr3*kgssgr5*kgssgr7*kgssgr9*kgssgr11 + kgssgr2*(kgssgr4*kgssgr6 +
kgssgr4*kgssgr7 + kgssgr5*kgssgr7)*kgssgr9*kgssgr11 +
X(20)*kgssgr2*kgssgr4*kgssgr6*kgssgr8*kgssgr12*X(39) +
X(21)*kgssgr3*kgssgr5*kgssgr7*kgssgr9*kgssgr12*X(39) + kgssgr2*(kgssgr4*kgssgr6 +
kgssgr4*kgssgr7 + kgssgr5*kgssgr7)*kgssgr9*kgssgr12*X(39) +
X(20)*X(21)*kgssgr3*kgssgr5*kgssgr7*kgssgr10*kgssgr12*X(39) +
X(20)*kgssgr2*(kgssgr4*kgssgr6 + kgssgr4*kgssgr7 +
kgssgr5*kgssgr7)*kgssgr10*kgssgr12*X(39) + X(20)^2*X(21)*kgssgr3*(kgssgr5 +
kgssgr6)*kgssgr8*kgssgr10*kgssgr12*X(39) + X(20)^2*(kgssgr2*kgssgr4 +
kgssgr2*kgssgr5 + kgssgr2*kgssgr6 +
kgssgr4*kgssgr6)*kgssgr8*kgssgr10*kgssgr12*X(39) +
X(20)*X(21)*kgssgr1*kgssgr3*kgssgr5*kgssgr7*kgssgr10*X(40) +
X(20)^2*kgssgr1*kgssgr4*kgssgr6*kgssgr8*kgssgr10*X(40) +
X(20)^2*X(21)*kgssgr1*kgssgr3*(kgssgr5 + kgssgr6)*kgssgr8*kgssgr10*X(40) +
X(20)*kgssgr1*kgssgr4*kgssgr6*kgssgr8*kgssgr11*X(40) +
X(20)*X(21)*kgssgr1*kgssgr3*(kgssgr5 + kgssgr6)*kgssgr8*kgssgr11*X(40) +
kgssgr1*(kgssgr4*kgssgr6 + kgssgr4*kgssgr7 +
kgssgr5*kgssgr7)*kgssgr9*kgssgr11*X(40) +
X(21)*kgssgr1*kgssgr3*(kgssgr5*kgssgr7*kgssgr9 + kgssgr5*kgssgr7*kgssgr11 +
kgssgr5*kgssgr9*kgssgr11 + kgssgr6*kgssgr9*kgssgr11 +
kgssgr7*kgssgr9*kgssgr11)*X(40));

vru5pe = VRu5PE*(kru5pe1*kru5pe3*X(49)/(kru5pe2 + kru5pe3) -
kru5pe2*kru5pe4*X(56)/(kru5pe2 + kru5pe3))/(1 + kru5pe1*X(49)/(kru5pe2 + kru5pe3)
+ kru5pe4*X(56)/(kru5pe2 + kru5pe3));

vr5pi = VR5PI*(-kr5pi2*kr5pi4*X(48)/(kr5pi2 + kr5pi3) +
kr5pi1*kr5pi3*X(49)/(kr5pi2 + kr5pi3))/(1 + kr5pi1*X(49)/(kr5pi2 + kr5pi3) +
kr5pi4*X(56)/(kr5pi2 + kr5pi3));

vtk1 = ktk1*X(51)*X(56) - ktk2*X(55);
vtk2 = ktk3*X(55) - ktk4*X(52)*X(18);
vtk3 = ktk5*X(52)*X(48) - ktk6*X(54);
vtk4 = ktk7*X(54) - ktk8*X(51)*X(50);
vtk5 = ktk9*X(52)*X(13) - ktk10*X(53);
vtk6 = ktk11*X(53) - ktk12*X(51)*X(15);

vta = VTA*((-X(13))*X(15)*kta2*kta4*kta6*kta8 +
X(18)*kta1*kta3*kta5*kta7*X(50))/(X(18)*(kta2 + kta3)*kta5*kta7 +
X(13)*kta2*kta4*(kta6 + kta7) + X(15)*X(18)*(kta2 + kta3)*kta5*kta8 + X(15)*(kta2
+ kta3)*kta6*kta8 + X(13)*X(15)*kta4*(kta2 + kta6)*kta8 + X(18)*kta1*kta5*(kta3 +
kta7)*X(50) + kta1*kta3*(kta6 + kta7)*X(50) + X(13)*kta1*kta4*(kta6 +
kta7)*X(50));

vak = 4300*(1 + Khadp/10^pH1 + Kkadv*k1)*X(1)*X(30) - 1400*(1 + Khamp/10^pH1 +
Kkamp*k1)*X(2)*X(31);

vatpase = katpase*X(31);
vox = kox*X(20);
voxnadh = koxNADH*X(38);
vlactransport = (1/Vi)*((-klactransporti)*Lace*Ve + klactransporti*Ve*X(28)/(1
+ 10^(-3.73 + pH1))*(1 + 10^(-3.73 + pH1)/rtv));
vpyrtransport = (1/Vi)*((-kpyrtransporti)*Pyre*Ve +
kpyrtransporti*Ve*X(47)/rtv);
vphostransport = (1/Vi)*((-kphostransporti)*Phose*Ve + kphostransporti*(10^(-

```

```

6.75 + pH1)/rtv^2 + 1/rtv)*Ve*X(46)/(1 + 10^(-6.75 + pH1)));
vmgatp = 2620.8*X(3)*(Kmgatp + Khatp*Kmghatp/10^pH1)*X(29)/(1 + Khatp/10^pH1 +
k1*Kkatp) - kmgatpd*X(31);
vmgadp = 1711.2*X(1)*(Kmgadp + Khadp*Kmghadp/10^pH1)*X(29)/(1 + Khadp/10^pH1 +
k1*Kkadp) - kmgadpd*X(30);
vmgb23pg = 2572.8*X(5)*(Kmgbpg + Khbpg*Kmghbpg/10^pH1)*X(29)/(1 + Khbpg/10^pH1
+ Kh2bpg*Khbpg/10^(2*pH1) + k1*Kkbpg + k1*Khbpg*Kkhhbpg/10^pH1) - kmgbpgd*X(33);
vmgb13pg = 729.6*X(4)*(Kmgbpg + Khbpg*Kmghbpg/10^pH1)*X(29)/(1 + Khbpg/10^pH1
+ Kh2bpg*Khbpg/10^(2*pH1) + k1*Kkbpg + k1*Khbpg*Kkhhbpg/10^pH1) - kmgb13pgd*X(32);
vmgfru16p2 = 3984*X(14)*(Kmgfru16p2 + Khf*Kmghf/10^pH1)*X(29)/(1 + Khf/10^pH1
+ Kh2f*Khf/10^(2*pH1) + k1*Kkf + k1*Khf*Kkhf/10^pH1) - kmgf16p2d*X(34);
vmgg1c16p2 = 3984*X(16)*(Kmgfru16p2 + Khf*Kmghf/10^pH1)*X(29)/(1 + Khf/10^pH1
+ Kh2f*Khf/10^(2*pH1) + k1*Kkf + k1*Khf*Kkhf/10^pH1) - kmgf16p2d*X(35);
vmgphos = (-kmgphosd)*X(36) + 40800*(1 + 6.30957344480193*10^(-8)*Khphos +
0.15*Kkphos)*X(29)*X(46)/(1 + Khphos/10^pH1 + k1*Kkphos);
vhbm gatp = (-X(27))*khbm gatpd + 46800*X(22)*(1 + 3.16978638492223*10^7*Kahb +
2.51188643150958*10^14*Kahb^2)*X(31)/(1 + 2^(1 + pH1)*5^pH1*Kahb +
10^(2*pH1)*Kahb^2);
vhbatp = -X(24)*khbatpd + 432000*X(3)*X(22)*(1 + 3.16978638492223*10^7*Kahb +
2.51188643150958*10^14*Kahb^2)/(1 + 2^(1 + pH1)*5^pH1*Kahb + 10^(2*pH1)*Kahb^2);
vhbadp = -X(23)*khbadpd + 300000*X(1)*X(22)*(1 + 3.16978638492223*10^7*Kahb +
2.51188643150958*10^14*Kahb^2)/(1 + 2^(1 + pH1)*5^pH1*Kahb + 10^(2*pH1)*Kahb^2);
vhbbpg = -X(26)*khbbpgd + 300000*X(5)*X(22)*(1 + 3.16978638492223*10^7*Kahb +
2.51188643150958*10^14*Kahb^2)/(1 + 2^(1 + pH1)*5^pH1*Kahb + 10^(2*pH1)*Kahb^2);
vhbb13pg = -X(25)*khbb13pgd + 380000*X(4)*X(22)*(1 +
3.16978638492223*10^7*Kahb + 2.51188643150958*10^14*Kahb^2)/(1 + 2^(1 +
pH1)*5^pH1*Kahb + 10^(2*pH1)*Kahb^2);
vsa = ksaf*(CONC_SA_max - X(57)) - ksar*X(31)*X(57); % reaction rate of
spectrin dissociation

% Rate of Change of Metabolite Concentrations (57 Metabolites)
dx(1) = -vak - vmgadp - vhadp;
dx(2) = vak;
dx(3) = -vmgatp - vhatp;
dx(4) = vgapdh - vpgk - vbpgsp1 - vmgb13pg - vhbb13pg;
dx(5) = vbpgsp7 - vmgb23pg - vhbbpg;
dx(6) = -vbpgsp1 + vbpgsp7 + vbpgsp9;
dx(7) = vbpgsp2 - vbpgsp3 - vbpgsp4 - vbpgsp8;
dx(8) = vbpgsp4 - vbpgsp6;
dx(9) = vbpgsp3 - vbpgsp5;
dx(10) = vbpgsp8 - vbpgsp9;
dx(11) = vbpgsp1 - vbpgsp2;
dx(12) = vbpgsp5 + vbpgsp6 - vbpgsp7;
dx(13) = -vtk5 + vta;
dx(14) = vpfk - vald - vmgfru16p2;
dx(15) = vgpi - vpfk + vtk6 + vta;
dx(16) = -vmgg1c16p2;
dx(17) = vkh - vgpi - vg6pdh;
dx(18) = vald - vtpi - vgapdh + vtk2 - vta;
dx(19) = vald + vtpi;
dx(20) = 2*vgssgr - 2*vox;
dx(21) = -vgssgr + vox;
dx(22) = -vhbm gatp - vhatp - vhadp - vhbbpg - vhbb13pg;

```



```

dx(23) = vhbap;
dx(24) = vhbap;
dx(25) = vhb13pg;
dx(26) = vhbpg;
dx(27) = vhbmgatp;
dx(28) = vldh + vldhp - vlactransport;
dx(29) = -vmgatp - vmgdp - vmgb23pg - vmgb13pg - vmgfru16p2 - vmgglc16p2 -
vmgphos;
dx(30) = vhk + vpfk - vpgk - vpk - vak + vatpase + vmgdp - vSA*(vSA < 0); %
(vSA < 0) term prevents reaction from proceeding in reverse direction (vSA
reaction not allowed to reattach links according to mathematical model)
dx(31) = -vhk - vpfk + vpgk + vpk + vak - vatpase + vmgatp - vhbmgatp +
vSA*(vSA < 0); % (vSA < 0) term prevents reaction from proceeding in reverse
direction (vSA reaction not allowed to reattach links according to mathematical
model)
dx(32) = vmgb13pg;
dx(33) = vmgb23pg;
dx(34) = vmgfru16p2;
dx(35) = vmgglc16p2;
dx(36) = vmgphos;
dx(37) = -vgapdh + vldh + voxnadh;
dx(38) = vgapdh - vldh - voxnadh;
dx(39) = vldhp - vg6pdh - vp6gdh + vgssgr;
dx(40) = -vldhp + vg6pdh + vp6gdh - vgssgr;
dx(41) = vpgm - veno - vbpgsp4;
dx(42) = vpgk - vpgm + vbpgsp2 - vbpgsp3;
dx(43) = vpglhydrolysis - vp6gdh;
dx(44) = vg6pdh - vpglhydrolysis;
dx(45) = veno - vpk;
dx(46) = -vgapdh - vbpgsp8 + 2*vbpgsp9 + vatpase - vphostransport - vmgphos;
dx(47) = vpk - vldh - vldhp - vpyrtransport;
dx(48) = vr5pi - vtk3;
dx(49) = vp6gdh - vru5pe - vr5pi;
dx(50) = vtk4 - vta;
dx(51) = -vtk1 + vtk4 + vtk6;
dx(52) = vtk2 - vtk3 - vtk5;
dx(53) = vtk5 - vtk6;
dx(54) = vtk3 - vtk4;
dx(55) = vtk1 - vtk2;
dx(56) = vru5pe - vtk1;
dx(57) = vSA;

end

```

### A.1.7 f\_plotting.m

```
function
f_plotting(nt,dt,U_LINKS,U_AREA_loc,U_AREA_g,U_VOLUME,U_BENDING,V_TOT,CON_t,CON_p,
r,r0,Fx,Xmin,Xmax,Ak,Elements,Links,LEN,CON,print_flag)
%
% Plots the results of a simulation of the Combined hrBC model
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hrBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   nt: number of time steps in simulation
%   dt: time step length
%   U_LINKS: vector of link potential energies wrt time
%   U_AREA_loc: vector of local area potential energies wrt time
%   U_AREA_g: vector of global area potential energies wrt time
%   U_VOLUME: vector of volume potential energies wrt time
%   U_BENDING: vector of bending potential energies wrt time
%   V_TOT: vector of total kinetic energies wrt time
%   CON_t: vector of connectivity wrt time
%   CON_p: vector of permanently broken links
%   r: array of node locations
%   r0: array of initial node locations
%   Fx: magnitude applied stretching force
%   Xmin: array of nodes with negative stretching force
%   Xmax: array of nodes with positive stretching force
%   Ak: vector of areas of individual elements
%   Elements: array of nodes that form elements
%   Links: array of nodes that form links
%   LEN: vector of link lengths
%   CON: vector of connectivity of links
%   print_flag: flag to print plot to .png files using export_fig
%
% OUTPUTS:
%   none
%
% GLOBAL VARIABLES:
%   A0t: equilibrium global area
%   L0: equilibrium link length
%   Ns: number of links
%   Nt: number of elements
%   Nv: number of nodes
%
% SAVE FILES:
```

```

% none

global A0t L0 Ns Nt Nv

% Pre-Plotting Calculations:
Nv = length(r);
Nt = length(Elements);
Ns = length(Links);
ELM_NUM = zeros(Nv,1);
ELM_LOC = zeros(Nv,7);
VERT_COL = zeros(Nv,1);
COL = zeros(Ns,3);
U_TOT = U_LINKS + U_AREA_loc + U_AREA_g + U_VOLUME + U_BENDING; % potential
energy [J]
E_TOT = U_TOT + V_TOT; % total energy [J]
nn = round(0.02*Nv); % number of nodes being stretched (in stretching
simulation)
tri_strain = (Ak - A0t/Nt)./(A0t/Nt); % area strains [unitless]
link_strain = (LEN - L0)./L0; % link strain [unitless]

for i = 1:1:Nv

    temp1 = transpose(find(Elements == i));
    ELM_NUM(i,1) = length(temp1);

    for j = 1:1:2

        for k = 1:1:ELM_NUM(i,1)

            if temp1(1,k) > Nt

                temp1(1,k) = temp1(1,k) - Nt;

            end

        end

    end

    ELM_LOC(i,1:ELM_NUM(i,1)) = temp1;

end

for i = 1:1:Nv

    temp2 = 0;

    for j = 1:1:ELM_NUM(i,1)

        temp2 = temp2 + tri_strain(ELM_LOC(i,j));

    end

end

```

```

        VERT_COL(i,1) = temp2/ELM_NUM(i,1);

    end

    MAP_LS = transpose(linspace(min(link_strain),max(link_strain),65));
    colormap(jet)
    cmp = colormap; close;

    for i = 1:1:Ns

        for j = 1:1:64

            if link_strain(i,1) >= MAP_LS(j,1) && link_strain(i,1) <= MAP_LS(j +
1,1)

                COL(i,:) = cmp(j,:);

            end

        end

    end

    end

    % Plotting:

    % 1) Links Energy
    f1 = figure(1);
    h11 = plot(dt:dt:dt*nt,U_LINKS);
    axis([0 dt*nt 0.99*min(U_LINKS) 1.01*max(U_LINKS)])
    a1 = gca;
    hXL1 = xlabel('Time [s]');
    hYL1 = ylabel('Energy [J]');
    hT1 = title({'Links Energy','(nt = ',num2str(nt),' , dt = ',num2str(dt),' [s],
Fx = ',num2str(Fx),' [N])' }));
    set(f1,'Color','w','MenuBar','none','Position',[400,138,1120,840])

    set(a1,'FontName','Helvetica','FontSize',12,'box','off','TickDir','out','TickLengt
h',[0.01 0.01],'XMinorTick','on','YMinorTick','on','YGrid','on','Linewidth',1)
    set([hXL1,hYL1],'FontName','Helvetica','FontSize',16)
    set(hT1,'FontName','Helvetica','FontSize',20)
    set(h11,'Color','b','Linewidth',2)

    % 2) Local Area Energy
    f2 = figure(2);
    h21 = plot(dt:dt:dt*nt,U_AREA_loc);
    axis([0 dt*nt 0.99*min(U_AREA_loc) 1.01*max(U_AREA_loc)])
    a2 = gca;
    hXL2 = xlabel('Time [s]');
    hYL2 = ylabel('Energy [J]');
    hT2 = title({'Local Area Energy','(nt = ',num2str(nt),' , dt = ',num2str(dt),'
[s], Fx = ',num2str(Fx),' [N])' }));
    set(f2,'Color','w','MenuBar','none','Position',[400,138,1120,840])

```

```

set(a2,'FontName','Helvetica','FontSize',12,'box','off','TickDir','out','TickLengt
h',[0.01 0.01],'XMinorTick','on','YMinorTick','on','YGrid','on','Linewidth',1)
set([hXL2,hYL2],'FontName','Helvetica','FontSize',16)
set(hT2,'FontName','Helvetica','FontSize',20)
set(h21,'Color','b','Linewidth',2)

% 3) Global Area Energy
f3 = figure(3);
h31 = plot(dt:dt:dt*nt,U_AREA_g);
axis([0 dt*nt 0.99*min(U_AREA_g) 1.01*max(U_AREA_g)])
a3 = gca;
hXL3 = xlabel('Time [s]');
hYL3 = ylabel('Energy [J]');
hT3 = title({'Global Area Energy',['(nt = ',num2str(nt),' dt = ',num2str(dt),' [s], Fx = ',num2str(Fx),' [N])']});
set(f3,'Color','w','MenuBar','none','Position',[400,138,1120,840])

set(a3,'FontName','Helvetica','FontSize',12,'box','off','TickDir','out','TickLengt
h',[0.01 0.01],'XMinorTick','on','YMinorTick','on','YGrid','on','Linewidth',1)
set([hXL3,hYL3],'FontName','Helvetica','FontSize',16)
set(hT3,'FontName','Helvetica','FontSize',20)
set(h31,'Color','b','Linewidth',2)

% 4) Volume Energy
f4 = figure(4);
h41 = plot(dt:dt:dt*nt,U_VOLUME);
axis([0 dt*nt 0.99*min(U_VOLUME) 1.01*max(U_VOLUME)])
a4 = gca;
hXL4 = xlabel('Time [s]');
hYL4 = ylabel('Energy [J]');
hT4 = title({'Volume Energy',['(nt = ',num2str(nt),' dt = ',num2str(dt),' [s], Fx = ',num2str(Fx),' [N])']});
set(f4,'Color','w','MenuBar','none','Position',[400,138,1120,840])

set(a4,'FontName','Helvetica','FontSize',12,'box','off','TickDir','out','TickLengt
h',[0.01 0.01],'XMinorTick','on','YMinorTick','on','YGrid','on','Linewidth',1)
set([hXL4,hYL4],'FontName','Helvetica','FontSize',16)
set(hT4,'FontName','Helvetica','FontSize',20)
set(h41,'Color','b','Linewidth',2)

% 5) Bending Energy
f5 = figure(5);
h51 = plot(dt:dt:dt*nt,U_BENDING);
axis([0 dt*nt 0.99*min(U_BENDING) 1.01*max(U_BENDING)])
a5 = gca;
hXL5 = xlabel('Time [s]');
hYL5 = ylabel('Energy [J]');
hT5 = title({'Bending Energy',['(nt = ',num2str(nt),' dt = ',num2str(dt),' [s], Fx = ',num2str(Fx),' [N])']});
set(f5,'Color','w','MenuBar','none','Position',[400,138,1120,840])

set(a5,'FontName','Helvetica','FontSize',12,'box','off','TickDir','out','TickLengt

```

```

h', [0.01 0.01], 'XMinorTick', 'on', 'YMinorTick', 'on', 'YGrid', 'on', 'Linewidth', 1)
set([hXL5, hYL5], 'FontName', 'Helvetica', 'FontSize', 16)
set(hT5, 'FontName', 'Helvetica', 'FontSize', 20)
set(h51, 'Color', 'b', 'Linewidth', 2)

% 6) Total Potential Energy
f6 = figure(6);
h61 = plot(dt:dt:dt*nt, U_TOT);
axis([0 dt*nt 0.99*min(U_TOT) 1.01*max(U_TOT)])
a6 = gca;
hXL6 = xlabel('Time [s]');
hYL6 = ylabel('Energy [J]');
hT6 = title(['Total Potential Energy', '(nt = ', num2str(nt), ', dt = ', num2str(dt), ' [s], Fx = ', num2str(Fx), ' [N])']);
set(f6, 'Color', 'w', 'MenuBar', 'none', 'Position', [400, 138, 1120, 840])

set(a6, 'FontName', 'Helvetica', 'FontSize', 12, 'box', 'off', 'TickDir', 'out', 'TickLength', 5)
h', [0.01 0.01], 'XMinorTick', 'on', 'YMinorTick', 'on', 'YGrid', 'on', 'Linewidth', 1)
set([hXL6, hYL6], 'FontName', 'Helvetica', 'FontSize', 16)
set(hT6, 'FontName', 'Helvetica', 'FontSize', 20)
set(h61, 'Color', 'b', 'Linewidth', 2)

% 7) Total Kinetic Energy
f7 = figure(7);
h71 = plot(dt:dt:dt*nt, V_TOT);
axis([0 dt*nt 0.99*min(V_TOT) 1.01*max(V_TOT)])
a7 = gca;
hXL7 = xlabel('Time [s]');
hYL7 = ylabel('Energy [J]');
hT7 = title(['Total Kinetic Energy', '(nt = ', num2str(nt), ', dt = ', num2str(dt), ' [s], Fx = ', num2str(Fx), ' [N])']);
set(f7, 'Color', 'w', 'MenuBar', 'none', 'Position', [400, 138, 1120, 840])

set(a7, 'FontName', 'Helvetica', 'FontSize', 12, 'box', 'off', 'TickDir', 'out', 'TickLength', 5)
h', [0.01 0.01], 'XMinorTick', 'on', 'YMinorTick', 'on', 'YGrid', 'on', 'Linewidth', 1)
set([hXL7, hYL7], 'FontName', 'Helvetica', 'FontSize', 16)
set(hT7, 'FontName', 'Helvetica', 'FontSize', 20)
set(h71, 'Color', 'b', 'Linewidth', 2)

% 8) Total Energy
f8 = figure(8);
h81 = plot(dt:dt:dt*nt, E_TOT);
axis([0 dt*nt 0.99*min(E_TOT) 1.01*max(E_TOT)])
a8 = gca;
hXL8 = xlabel('Time [s]');
hYL8 = ylabel('Energy [J]');
hT8 = title(['Total Energy', '(nt = ', num2str(nt), ', dt = ', num2str(dt), ' [s], Fx = ', num2str(Fx), ' [N])']);
set(f8, 'Color', 'w', 'MenuBar', 'none', 'Position', [400, 138, 1120, 840])

set(a8, 'FontName', 'Helvetica', 'FontSize', 12, 'box', 'off', 'TickDir', 'out', 'TickLength', 5)
h', [0.01 0.01], 'XMinorTick', 'on', 'YMinorTick', 'on', 'YGrid', 'on', 'Linewidth', 1)
set([hXL8, hYL8], 'FontName', 'Helvetica', 'FontSize', 16)

```

```

set(hT8,'FontName','Helvetica','FontSize',20)
set(h81,'Color','b','Linewidth',2)

% 9) Connectivity
f9 = figure(9);
h91 = plot(dt:dt:dt*nt,CON_t);
axis([0 dt*nt 0 1])
a9 = gca;
hXL9 = xlabel('Time [s]');
hYL9 = ylabel('Connectivity [unitless]');
hT9 = title({'Connectivity','(nt = ',num2str(nt),' , dt = ',num2str(dt),' [s],
Fx = ',num2str(Fx),' [N])''));
set(f9,'Color','w','MenuBar','none','Position',[400,138,1120,840])

set(a9,'FontName','Helvetica','FontSize',12,'box','off','TickDir','out','TickLength',[0.01 0.01],
'XMinorTick','on','YMinorTick','on','YGrid','on','Linewidth',1)
set([hXL9,hYL9],'FontName','Helvetica','FontSize',16)
set(hT9,'FontName','Helvetica','FontSize',20)
set(h91,'Color','b','Linewidth',2)

% 10) hRBC Final Configuration (With Attachment Point Nodes Highlighted)
f10 = figure(10);
hold on
h101 = trimesh(Elements,r(:,1),r(:,2),r(:,3),'EdgeColor','red'); %%ok<NASGU>

for i = 1:1:nn

    plot3(r(Xmin(i,1),1),r(Xmin(i,1),2),r(Xmin(i,1),3),'.k','MarkerSize',20);
    plot3(r(Xmax(i,1),1),r(Xmax(i,1),2),r(Xmax(i,1),3),'.k','MarkerSize',20);

end

hold off
view(3)
axis equal off
hT10 = title({'hRBC Final Configuration','(With Attachment Point Nodes
Highlighted)'});
set(f10,'Color','w','MenuBar','none','Position',[400,138,1120,840])
set(hT10,'FontName','Helvetica','FontSize',20)

% 11) hRBC Final Configuration (With Initial Configuration)
f11 = figure(11);
hold on
h111 = trimesh(Elements,r(:,1),r(:,2),r(:,3),'EdgeColor','red');
h112 = plot3(r0(:,1),r0(:,2),r0(:,3),'.k');
hold off
view(3)
axis equal off
hT11 = title({'hRBC Final Configuration','(With Initial Configuration)'});
set(f11,'Color','w','MenuBar','none','Position',[400,138,1120,840])
set(hT11,'FontName','Helvetica','FontSize',20)
set(h111,'FaceAlpha',0.3)
set(h112,'MarkerSize',10)

```

```

% 12) hRBC Final Configuration (With Dissociations)
f12 = figure(12);
hold on
h121 = trimesh(Elements,r(:,1),r(:,2),r(:,3),'EdgeColor','red'); %%ok<NASGU>

for i = 1:1:Ns

    if CON(i,1) == 0

        line([r(Links(i,1),1) r(Links(i,2),1)],[r(Links(i,1),2)
r(Links(i,2),2)],[r(Links(i,1),3) r(Links(i,2),3)],'Color','g','Linewidth',2)

    end

    if CON_p(i,1) == 0

        line([r(Links(i,1),1) r(Links(i,2),1)],[r(Links(i,1),2)
r(Links(i,2),2)],[r(Links(i,1),3) r(Links(i,2),3)],'Color','k','Linewidth',2)

    end

end

hold off
view(3)
axis equal off
hT12 = title({'hRBC Final Configuration','(With Dissociations)'});
set(f12,'Color','w','MenuBar','none','Position',[400,138,1120,840])
set(hT12,'FontName','Helvetica','FontSize',20)

% 13) hRBC Final Configuration (With Vertex Strains)
f13 = figure(13);
h131 =
trisurf(Elements,r(:,1),r(:,2),r(:,3),'FaceVertexCData',VERT_COL,'FaceColor','interp');
colorbar
caxis([min(tri_strain) max(tri_strain)])
view(3)
axis equal off
hT13 = title({'hRBC Final Configuration','(With Vertex Strains)'});
set(f13,'Color','w','MenuBar','none','Position',[400,138,1120,840])
set(hT13,'FontName','Helvetica','FontSize',20)
set(h131,'EdgeAlpha',0)

% 14) hRBC Final Configuration (With Area Strains)
f14 = figure(14);
h141 = trisurf(Elements,r(:,1),r(:,2),r(:,3),tri_strain); %%ok<NASGU>
colorbar
caxis([min(tri_strain) max(tri_strain)])
view(3)
axis equal off
hT14 = title({'hRBC Final Configuration','(With Area Strains)'});

```



```

set(f14,'Color','w','MenuBar','none','Position',[400,138,1120,840])
set(hT14,'FontName','Helvetica','FontSize',20)

% 15) hRBC Final Configuration (With Link Strains)
f15 = figure(15);
set(f15,'Renderer','OpenGL')
hold on

for i = 1:1:Ns

    line([r(Links(i,1),1) r(Links(i,2),1)], [r(Links(i,1),2)
r(Links(i,2),2)], [r(Links(i,1),3) r(Links(i,2),3)], 'color', COL(i,:))

end

hold off
colorbar
caxis([min(link_strain) max(link_strain)])
view(3)
axis equal off
hT15 = title({'hRBC Final Configuration', '(With Link Strains)'});
set(f15,'Color','w','MenuBar','none','Position',[400,138,1120,840])
set(hT15,'FontName','Helvetica','FontSize',20)

% Saving Plots:
if print_flag == 1

    export_fig(f1,'1_Links_Energy.png','-q101','-painters')
    export_fig(f2,'2_Local_Area_Energy.png','-q101','-painters')
    export_fig(f3,'3_Global_Area_Energy.png','-q101','-painters')
    export_fig(f4,'4_Volume_Energy.png','-q101','-painters')
    export_fig(f5,'5_Bending_Energy.png','-q101','-painters')
    export_fig(f6,'6_Total_Potential_Energy.png','-q101','-painters')
    export_fig(f7,'7_Total_Kinetic_Energy.png','-q101','-painters')
    export_fig(f8,'8_Total_Energy.png','-q101','-painters')
    export_fig(f9,'9_Connectivity.png','-q101','-painters')
    export_fig(f10,'10_hRBC_Final.png','-q101')
    export_fig(f11,'11_hRBC_Final.png','-q101')
    export_fig(f12,'12_hRBC_Final.png','-q101')
    export_fig(f13,'13_hRBC_Final.png','-q101')
    export_fig(f14,'14_hRBC_Final.png','-q101')
    export_fig(f15,'15_hRBC_Final.png','-q101')

end

end

```

## A.1.8 f\_pp.m

```
function
[ts_len,ts_ratio,t_sim_t,U_AVG,V_AVG,E_AVG,DT,DA,EI,r_deg,dL,pdeg6,Qa,Qh,RSA] =
f_pp(r,t_end,nt,dt,Elements,Links,LEN,U_TOT,V_TOT)
%
% Performs post-processing calculations after a simulation of the Combined
% hRBC model
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hrBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   r: array of node locations
%   t_end: wall time of simulation
%   nt: number of time steps
%   dt: length of time step
%   Elements: array of nodes that form elements
%   Links: array of nodes that form links
%   LEN: vector of link lengths
%   U_TOT: vector of potential energies of model
%   V_TOT: vector of kinetic energies of model
%
% OUTPUTS:
%   ts_len: wall time per time step
%   ts_ratio: ratio of wall time per time step to length of time step
%   t_sim: total simulation time elapsed
%   U_AVG: time-averaged potential energy
%   V_AVG: time-averaged kinetic energy
%   E_AVG: time-averaged total energy
%   DT: transverse diameter
%   DA: axial diameter
%   EI: elongation index
%   r_deg: table of vertex degree results
%   dL: distribution of link lengths
%   pdeg6: percentage of degree 6 nodes
%   Qa: arithmetic mean of triangulation quality measure
%   Qh: harmonic mean of triangulation quality measure
%   RSA: relative shape anisotropy
%
% GLOBAL VARIABLES:
%   Nt: number of elements
%
% SAVE FILES:
%   none
```

```

global Nt

% Simulation Time Calculations:
ts_len = t_end/nt; % wall time per time step [s]
ts_ratio = ts_len/dt; % ratio of wall time per time step to length of time
step [unitless]
t_sim_t = dt*nt; % total simulation time elapsed [s]

% Time Average Energy Calculations:
U_AVG = mean(U_TOT); % time-average potential energy at end of simulation [J]
V_AVG = mean(V_TOT); % time-average kinetic energy at end of simulation [J]
E_AVG = mean(U_TOT + V_TOT); % time-average total energy at end of simulation
[J]

% Elongation Index Calculations:
Dt = zeros(length(r),1); % initialize nodal radius vector

cx = mean(r(:,1)); % x-coordinate of center of mass [m]
cy = mean(r(:,2)); % y-coordinate of center of mass [m]
cz = mean(r(:,3)); % z-coordinate of center of mass [m]

for i = 1:length(r)

    Dt(i,1) = sqrt((r(i,2) - cy)^2 + (r(i,3) - cz)^2); % nodal distance (from
center of mass)

end

DT = 2*max(Dt); % transverse diameter (smaller diameter) [m]
DA = max(r(:,1)) - min(r(:,1)); % axial diameter (larger diameter) [m]
EI = (DA - DT)/(DA + DT); % elongation index [unitless]

% Triangulation Quality Calculations:

% Vertex Degree Quality
bc = histc(Links,unique(Links)); % count number of times each node appears in
Links
deg = bc(:,1) + bc(:,2); % degree of each node
dn = histc(deg,unique(deg)); % number of nodes of each degree
r_deg = [unique(deg) dn]; % table of vertex degree results

min_deg = min(deg); %%ok<NASGU> % minimum degree present
max_deg = max(deg); %%ok<NASGU> % maximum degree present
avg_deg = mean(deg); %%ok<NASGU> % average degree of all nodes
std_deg = std(deg); %%ok<NASGU> % standerd deviation of all node degrees
avg_len = mean(LEN); % average length of all edges
std_len = std(LEN); % standerd deviation of all edge lengths
dL = std_len/avg_len; % distribution of link length
pdeg6 = dn(unique(deg) == 6)/sum(dn); % percentage of degree 6 nodes

% Element Area Quality
Q = zeros(Nt,1); % initialize element quality vector

```

```

for i = 1:1:Nt

    a = Elements(i,1);
    b = Elements(i,2);
    c = Elements(i,3);

    xa = r(a,1); % x-coordinate of node a
    ya = r(a,2); % y-coordinate of node a
    za = r(a,3); % z-coordinate of node a
    xb = r(b,1); % x-coordinate of node b
    yb = r(b,2); % y-coordinate of node b
    zb = r(b,3); % z-coordinate of node b
    xc = r(c,1); % x-coordinate of node c
    yc = r(c,2); % y-coordinate of node c
    zc = r(c,3); % z-coordinate of node c

    rab = [(xa - xb) (ya - yb) (za - zb)];
    rbc = [(xb - xc) (yb - yc) (zb - zc)];
    rca = [(xc - xa) (yc - ya) (zc - za)];

    Lab = sqrt(rab(1,1)*rab(1,1) + rab(1,2)*rab(1,2) + rab(1,3)*rab(1,3));
    Lbc = sqrt(rbc(1,1)*rbc(1,1) + rbc(1,2)*rbc(1,2) + rbc(1,3)*rbc(1,3));
    Lca = sqrt(rca(1,1)*rca(1,1) + rca(1,2)*rca(1,2) + rca(1,3)*rca(1,3));

    Q(i,1) = (Lbc + Lca - Lab)*(Lca + Lab - Lbc)*(Lab + Lbc -
Lca)/(Lab*Lbc*Lca); % triangle quality
    % % NOTE: Q >= 0.5 indicates triangle is of acceptable quality
    % % max(Q) = 1 (for an equilateral triangle)
    % % min(Q) = 0 (for a degenerate triangle)

end

Qa = sum(Q)/Nt; % arithmetic mean quality
Qh = Nt/sum(1./Q); % harmonic mean quality

% Gyration Tensor Calculations
Gyr = zeros(3,3);
Nv = length(r);
rx = r(:,1) - cx;
ry = r(:,2) - cy;
rz = r(:,3) - cz;
Gyr(1,1) = sum(rx.*rx)/Nv;
Gyr(1,2) = sum(rx.*ry)/Nv;
Gyr(1,3) = sum(rx.*rz)/Nv;
Gyr(2,1) = sum(ry.*rx)/Nv;
Gyr(2,2) = sum(ry.*ry)/Nv;
Gyr(2,3) = sum(ry.*rz)/Nv;
Gyr(3,1) = sum(rz.*rx)/Nv;
Gyr(3,2) = sum(rz.*ry)/Nv;
Gyr(3,3) = sum(rz.*rz)/Nv;
[eVect,eVals] = eig(Gyr); %#ok<ASGLU>

```

```

L1 = evals(1,1);
L2 = evals(2,2);
L3 = evals(3,3);
Rg = sqrt(L1^2 + L2^2 + L3^2); %#ok<NASGU>
asphericity = L3^2 - 0.5*(L1^2 + L2^2); %#ok<NASGU>
acylindricity = abs(L1^2 - L2^2); %#ok<NASGU>
RSA = sqrt((3/2)*((L1^4 + L2^4 + L3^4)/(L1^2 + L2^2 + L3^2)^2) - 1/2); %
relative shape anisotropy

end

```

## A.1.9 f\_save.m

```
function [DATE_END] = f_save(save_flag,alert_case)
%
% Saves data from simulation of the Combined hRBC model to .mat file
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   hRBC_Stretching_Simulation.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   save_flag: flag for saving data
%             (0 = data is not saved, 1 = data is saved)
%   alert_case: successful or unsuccessful completion
%             (0 = unsuccessful completion, 1 = successful completion)
%
% OUTPUT:
%   DATE_END: date and time of end of simulation
%
% GLOBAL VARIABLES:
%   none
%
% SAVE FILE:
%   DATA_hRBC_Stretching_Simulation... .mat
%           -or-
%   ERROR_hRBC_Stretching_Simulation... .mat

DATE_END = datestr(now,'mm_dd_yyyy__HH_MM_SS'); % date and time of simulation
completion

% Collect Variables from Workspace
TEMP_2 = evalin('caller','whos'); % record all variables from workspace

for i = 1:length(TEMP_2)

    TEMP_3 = evalin('caller',[TEMP_2(i).name ';' ]); %#ok<NASGU> % get variable
name
    eval([TEMP_2(i).name,'= TEMP_3;']); % assign value to variable name

end

clear TEMP_2 TEMP_3 i

% Save Variables in .mat File
if save_flag == 1
```

```

        if alert_case == 0 % if unsuccessful completion

            FILE_NAME =
['ERROR_hrBC_Stretching_Simulation__',num2str(nt,2),'__',num2str(dt,2),'__',DATE_E
ND]; % file name

            elseif alert_case == 1 % if successful completion

                FILE_NAME =
['DATA_hrBC_Stretching_Simulation__',num2str(nt,2),'__',num2str(dt,2),'__',DATE_EN
D]; % file name

            end

        save([FILE_NAME, '.mat'], 'DATE_START', 'DATE_END', 'NOTE', 'A0t', 'Ak', 'AREA', 'CON', 'CO
N_p', 'CON_t', 'CONC', 'Elements', 'Fx', 'dt', 'L0', 'Links', 'LEN', 'Lm', 'n_error', 'nt', 'n
um_dis', 'r', 'r0', 't_end', 'U_LINKS', 'U_AREA_loc', 'U_AREA_g', 'U_VOLUME', 'U_BENDING',
'U_TOT', 'V_TOT', 'VOLUME', 'vel', 'xmax', 'xmin')

        end

    end
end

```

### A.1.10 hRBC\_Parameter\_Derivation.m

```
% hRBC_Parameter_Derivation
%
% Calculates all the parameters of the hRBC model that are used in
% simulations and saves the results in a .mat file.
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   none
%
% LOAD FILES:
%   hRBC_Triangulation.mat
%
% INPUTS:
%   none
%
% OUTPUTS:
%   none
%
% GLOBAL VARIABLES:
%   none
%
% SAVE FILES:
%   hRBC_Parameters.mat

clear all
close all
clc

% Load Data:
load('hRBC_Triangulation.mat') % triangulation and initial node locations

% Given Parameters:
% Constants
Na = 6.0221415e23; % Avogadro constant [1/mol]
kB = 1.380653e-23; % Boltzmann constant [J/K]
T = 23 + 273.15; % temperature [K]
kBT = kB*T;

% hRBC
D0 = 7.82e-6; % diameter [m]
rho_w = 993.68; % density of water [kg/m^3]
rho_hRBC = 1.15*rho_w; % density of hRBC [kg/m^3]
A0t = 135e-12; % equilibrium global area [m^2]
V0t = 94e-18; % equilibrium volume [m^3]
u0 = 6.3e-6; % observed linear elastic shear modulus [N/m]
```



```

% % NOTE:
% % Shear deformation is area-preserving, therefore only spring forces
% % contribute to the membrane shear modulus

% WLC
x0 = 1/2.2; % L0/Lm ratio [unitless]

% POW
m = 2; % power law exponent [unitless]
kc = 2.4e-19; % bending rigidity [J]

% LJ
S = 22 + 17; % number of links in spectrin chain (imaginary quantity) [unitless]
Ka = 1e12; % equilibrium association constant of S-A bond [unitless]

% Dissociation
wt = 1/(1e7/5); % wait-time (before link reassociation) [s]
% % NOTE:
% % wt = 1/k_d, where k_d is the rate of link dissociation

% Viscosity
no = 1.2e-3; % blood plasma viscosity [N s/m^2]
ni = 5*no; % cytoplasm viscosity [N s/m^2]
nm = 22e-3; % membrane viscosity [N s/m^2]
ma = 6.974263464e-23; % mass of Actin node [kg]
rho_a = 1.41e3; % density of Actin node [kg/m^3]

% Derived Parameters:
% WLC-POW
Nt = 2*Nv - 4; % number of triangles (elements)
Ns = 3*Nt/2; % number of sides (links)
A0 = A0t/Nt; % average area of individual triangle [m^2]
L0_eff = sqrt(4*A0/sqrt(3)); % effective equilibrium length [m]
up = 2*u0; % actual linear elastic shear modulus [N/m]
% % NOTE:
% % enforcing condition that u0 = up/2 due to partially connected network

Y = 3.92453*up; % Young's modulus [N/m]
K = (Y*up)/(4*up - Y); % Area-compression modulus (2-D Bulk modulus) [N/m]
v = (K - up)/(K + up); % Poisson's ratio (since 2-D, v = 1 gives a incompressible
material, instead of v = 0.5) [unitless]
Lm = L0_eff/x0; % maximum length of spectrin extension (contour length) [m]
kb = 2*kc/sqrt(3); % bending stiffness [J]
theta0 = acos((sqrt(3)*(Nv - 2) - 5*pi)/(sqrt(3)*(Nv - 2) - 3*pi)); % equilibrium
angle [rad]

syms Lp kp
eqn1 = (sqrt(3)*kBT)/(4*Lp*Lm*x0)*(x0/(2*(1 - x0)^3) - 1/(4*(1 - x0)^2) + 1/4) +
sqrt(3)*kp*(m + 1)/(4*L0_eff^(m + 1)); % symbolic expression for WLC-POW shear
modulus at effective equilibrium length, f(Lp,kp)
Lp = eval(solve(eqn1 - u0,Lp)); % symbolic expression for persistence length
eqn2 = kp/(L0_eff^m) - (kBT/Lp)*(1/(4*(1 - x0)^2) - 1/4 + x0); % symbolic

```

```

expression for spring forces at effective equilibrium length, f(kp)
kp = eval(solve(eqn2,kp)); % POW spring constant [N m^2]
Lp_p = eval(Lp); % original persistence length [m]
clear Lp
syms L Lp
xn = L/Lm; % normalized extension [unitless]
eqn3 = (sqrt(3)*kBT)/(4*Lp*L)*((L/Lm)/(2*(1 - (L/Lm))^3) - 1/(4*(1 - (L/Lm))^2) + 1/4) + sqrt(3)*kp*(m + 1)/(4*L^(m + 1)); % symbolic expression for WLC-POW shear modulus, f(L,Lp)
eqn4 = (kBT/kp)*(L^m)*(1/(4*(1 - xn)^2) - 1/4 + xn); % symbolic expression for persistence length, f(L)
eqn5 = subs(eqn3,Lp,eqn4); % symbolic expression for WLC-POW shear modulus, f(L)
val = solve(eqn5 - up,L); % symbolic expression for equilibrium length
L0 = eval(val(2,1)); % equilibrium length [m]
clear val
Lp = subs(eqn4,L,L0); % persistence length [m]
Lavg = sqrt(2*Lp*Lm - 2*(Lp^2)*(1 - exp(-Lm/Lp))); % average end-to-end spring length [m] (from polymer dynamics)
clear L

% LJ
r0 = Lm/(S - 1); % equilibrium length of links in spectrin chain [unitless]
Eng = kB*T*log(Ka); % association energy of S-A bond [J]
sig = 2*r0/(2^(1/6)); % characteristic interaction length scale [m]
xmin = sig*(26/7)^(1/6); % location of minimum of LJ force curve (point of inflection on LJ energy curve) [m]
eff_a = L0 - 2*r0; % effective length #1 [m]
xmin1 = sig*(26/7)^(1/6) + eff_a; % location of minimum of LJ force (when shifted to align with L0 for Ft1) [m]

syms L
Flj = 24*Eng*(2*(sig^12)/((L - eff_a)^13) - (sig^6)/((L - eff_a)^7)); % attractive LJ force (shifted to align with L0 for Ft1) [N]
Fmin = subs(Flj,L,xmin1); % magnitude of force required to break S-A bond (minimum of LJ force curve) [N]
Fpow = kp/(L^m); % repulsive POW force [N]
xn = L/Lm; % normalized length [unitless]
Fwlc = -(kB*T/Lp)*(1/(4*(1 - xn)^2) - 1/4 + xn); % attractive WLC force [N]
F0 = Fpow + Fwlc; % total force in original model [N]
xmin2 = solve(F0 - Fmin,L); % symbolic expression for location of maximum attractive force (before spectrin filament "breaks" due to S-A bond) on original model [m]
xcut = eval(xmin2(3,1)); % location of maximum attractive force (before spectrin filament "breaks" due to S-A bond) on original model [m]
adjx = xcut - xmin; % effective length #2 [m]

% Scaling Parameters:
% Necessary because parameters given in Fedosov are non-dimensionalized

D0_M = 15.87; % scaled diameter (for Nv_M = 27344) [unitless]
Y0_M = 392.453; % scaled Young's modulus [unitless]
ka_M = 4900; % scaled global area constraint constant [unitless]

```

```

kd_M = 100; % scaled local area constraint constant [unitless]
kv_M = 5000; % scaled volume constraint constant [unitless]

r_M = D0/D0_M; % length scaling parameter [m]
kBT_M = (Y/Y0_M)*(D0/D0_M)^2; % energy scaling parameter [J]
N_M = kBT_M/r_M; % force scaling parameter [N]

% Solve for Other Spring Parameters:
% Using Scaling Parameters
ka = ka_M*N_M/r_M;
kd = kd_M*N_M/r_M;
kv = kv_M*N_M/(r_M^2);

% Save Parameters:
DATE_hrBC_Parameters = datestr(now,'mm_dd_yyyy__HH_MM_SS'); % date and time of
completion of parameter derivation
save('hrBC_Parameters.mat','DATE_hrBC_Parameters','A0','A0t','adjx','Eng','kB','ka',
'kb','kd','kp','kv','L0','Lm','Lp','m','ma','Na','Ns','Nt','Nv','ni','nm','no','
rho_a','rho_hrBC','sig','T','theta0','v0t','x0','xcut','wt')

```

### A.1.11 hRBC\_Stretching\_Simulation.m

```
% hRBC_Stretching_Simulation
%
% Performs a stretching simulation of the Combined hRBC model
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   none
%
% LOAD FILES:
%   hRBC_Triangulation.mat
%   hRBC_Parameters.mat
%
% INPUTS:
%   none
%
% OUTPUTS:
%   none
%
% GLOBAL VARIABLES:
%   A0: equilibrium local area
%   A0t: equilibrium global area
%   adjx: effective length in LJ
%   CONC_SA_max:
%   Eng: association energy of spectrin-actin bond
%   kB: Boltzmann constant
%   ka: global area constraint
%   kb: bending constraint
%   kd: local area constraint
%   kp: POW constraint
%   kv: volume constraint
%   L0: equilibrium length of links
%   Lm: contour length of links
%   Lp: persistence length of links
%   m: exponent in POW
%   ma: mass of an actin node
%   Na: Avogadro constant
%   Ns: number of links
%   Nt: number of triangles
%   Nv: number of nodes
%   ni: viscosity of cytoplasm
%   nm: viscosity of cell membrane
%   no: viscosity of blood plasma
%   rho_a: density of an actin node
%   sig: characteristic interaction length scale in LJ
%   T: temperature
%   theta0:
%   V0t: equilibrium volume
```

```

% VOLUME: actual volume
% x0: ratio of equilibrium length of links to contour length of links
% xcut: location of maximum attractive force in LJ
%
% SAVE FILES:
% none (see f_save.m)

clear all
close all
clc

% Pre-Simulation Setup:
DATE_START = datestr(now,'mm_dd_yyyy__HH_MM_SS'); % date and time of simulation
initiation

global A0 A0t adjx CONC_SA_max Eng kB ka kb kd kp kv L0 Lm Lp m ma Na Ns Nt Nv ni
nm no rho_a sig T theta0 V0t VOLUME x0 xcut

load('hrBC_Parameters.mat') % parameters
load('hrBC_Minimized.mat','Elements','Links','bending_pts','r','CON','CONC','LEN',
'VOLUME') % triangulation and initial conditions

% Flags (0 = off, 1 = on):
save_flag = 0; % flag for saving variables
SS_flag = 1; % flag for running stretching simulation
plotting_flag = 0; % flag for plotting results of stretching simulation
print_flag = 0; % flag for saving plots

% Basic Simulation Parameters:
nt = 2e4; % number of time steps [unitless]
dt = 2.5e-8; % length of time step [s]
Fx = 50e-12; % magnitude of applied force [N]
NOTE = ''; % details about simulation (for saving)

nwt = round(wt/dt); % maximum number of time steps in wait time counter [unitless]
Lam = 0.63; % verlet integration term [unitless]
MASS = rho_hrBC*VOLUME/Nv; % nodal mass of hrBC [kg]
r0 = r; % record initial positions [m]

options = odeset('AbsTol',1e-12,'RelTol',1e-12,'MassSingular','no'); % ode15s
options

% Initialize/Declare Arrays/Vectors/Terms:
vel = zeros(Nv,3); % initialize current velocity array to zero [m/s]
acc = zeros(Nv,3); % initialize current acceleration array to zero [m/s^2]
vel_p = zeros(Nv,3); % initialize previous velocity array to zero [m/s]

U_LINKS = zeros(nt,1); % initialize in-plane potential energy vector [J]
U_AREA_loc = zeros(nt,1); % initialize local area potential energy vector [J]
U_AREA_g = zeros(nt,1); % initialize global area potential energy vector [J]
U_VOLUME = zeros(nt,1); % initialize volume potential energy vector [J]
U_BENDING = zeros(nt,1); % initialize bending potential energy vector [J]

```

```

U_TOT = zeros(nt,1); % initialize total potential energy vector [J]
V_TOT = zeros(nt,1); % initialize total kinetic energy vector [J]
E_TOT = zeros(nt,1); % initialize total energy vector [J]

F_C = zeros(Nv,3); % initialize conservative force array
F_D = zeros(Nv,3); % initialize dissipative force array
F_TOT = zeros(Nv,3); % initialize total force array
F_TOT_p = zeros(Nv,3); % initialize previous total force array
F_APP = zeros(Nv,3); % array of applied force

CON_p = ones(Ns,1); % initialize permanent connectivity vector
NUM = transpose(1:1:Ns); % declare vector of numerics
V_links = zeros(Ns,1); % initialize selected links vector
WT = zeros(Ns,1); % initialize wait time counter vector
CON_t = zeros(nt,1); % initialize connectivity as a function of time vector

for i = 1:1:Ns

    if CON(i,1) == 0

        WT(i,1) = randi(nwt,1); % set wait time counters of initially disconnected
links to values selected from uniform random distribution

    end

end

WEIGHTS = LEN; % vector of selection weights, based on current stretch of links
WEIGHTS(CON == 0) = 0; % set selection weights of initially disconnected links to
0
weights = WEIGHTS./sum(WEIGHTS); % normalized vector of selection weights
VL = VOLUME*1e3; % update volume [L]
num_avail = Ns; % initial number of links available to dissociate
CONC_SA_max = (num_avail/Na)/VL; % set current maximum concentration of links
[mo1/L], [M]

% Setup Stretching Parameters:
nn = round(0.02*Nv); % number of nodes force is applied to
Xv(:,1) = r(:,1);
[~,sI] = sort(Xv);
Imin = sI;
Imax = flipud(sI);
Xmin = Imin(1:nn,1); % indicies of nodes negative force is applied to
Xmax = Imax(1:nn,1); % indicies of nodes positive force is applied to

if ss_flag == 1

    for i = 1:1:nn

        F_APP(Xmin(i,1),1) = -Fx/nn;
        F_APP(Xmax(i,1),1) = Fx/nn;

    end

end

```

```

else

    Fx = 0;

end

clear Xv sI Imax Imin

% Simulation:
tstart = tic; % initialize start time of simulation (wall time)

for t = 1:1:nt

    % Update Numerical Integration Time Span:
    tb = t*dt - dt; % (time before) time at beginning of time step [s]
    ta = t*dt; % (time after) time at end of time step [s]

    % Break Overextended Links:
    CON_p((LEN > Lm) & (CON == 1)) = 0; % break currently connected links with
with a length > Lm
    CON(CON_p == 0) = 0; %ok<SAGROW> % set connectivity of links permanently to
zero
    num_avail = sum(CON_p); % update maximum number of links available to
reconnect
    CONC_SA_max = (num_avail/Na)/VL; %ok<NASGU> % update current maximum
concentration of links [mol/L], [M]

    % Reconnect Links and Update Wait Time Counters:
    num_reas = sum(WT == 1); % number of links reassociating this time step
[unitless]
    WT(WT > 0) = WT(WT > 0) - 1; % decrement number of timesteps left in wait time
counter by one
    WT(WT < 0) = 0; % set wait time counters less than zero back to zero (should
only happen due to an error)
    CON(WT == 0) = 1; %ok<SAGROW> % update connectivity of reassociated links

    % Update Selection Weights:
    WEIGHTS(WT == 0) = LEN(WT == 0); % update vector of selection weights to
current link lengths

    WEIGHTS(LEN > Lm) = 0; % set selection weights of links with a length > Lm to
zero
    weights = WEIGHTS./sum(WEIGHTS); % update normalized vector of positive
weights

    % Solve for Current Metabolite Concentrations:
    [~,X] = ode15s(@f_metabolism_mex,[tb ta],CONC,options);
    CONC = transpose(X(end,:)); % update current metabolite concentrations
[mol/L], [M]
    clear X % clear array of metabolite concentrations during timestep [mol/L],

```

```

[M]

% Solve for Number of Links to Dissociate:
num_dis = round(sum(CON) - floor(CONC(57,1)*Na*VL)); % number of links to
dissociate in current timestep [unitless]

if ~(isscalar(num_dis) && isnumeric(num_dis) && (num_dis == round(num_dis))) %
error if number of nodes to dissociate greater than threshold value

    n_error = t; %#ok<NASGU> % record timestep where error occurred
    f_save(save_flag,0); % save data to error dump
    error('Error: Invalid number of nodes to dissociate');

end

num_dis(num_dis < 0) = 0; % set number of links dissociated in current
timestep < 0 to 0 (for datasample, num_dis must be >= 0) [unitless]

% Select Links to Dissociate and Dissociate Links:
V_links(1:num_dis,1) =
datasample(NUM,num_dis,'Replace',false,'weights',weights); % selection of links to
dissociate in current timestep [unitless]
[WT,CON,WEIGHTS] = f_diss_mex(WT,CON,WEIGHTS,num_dis,nwt,V_links); %
dissociates selected links
V_links = zeros(Ns,1); % re-initialize selected links vector

if sum(CON) < 0.3*num_avail % error if connectivity is less than threshold
value

    n_error = t; %#ok<NASGU> % record timestep where error occurred
    f_save(save_flag,0); % save data to error dump
    error('Error: Invalid connectivity');

end

% Solve for New Positions and Intermediate Velocities Based on Current Forces:
MASS = rho_hRBC*VOLUME/Nv; % update mass per node [kg]
acc = F_TOT./MASS; % current accelerations [m/s^2]
r = r + vel.*dt + 0.5.*acc.*(dt^2); % new positions [m]
vel_p = vel; % update previous velocities [m/s]
vel = vel + Lam.*acc.*dt; % intermediate velocities [m/s]

% Solve for Forces and Potential Energies:
[F_C,U1,U2,U3,LEN,Ak,AREA] =
f_cons_forces_mex(Links,Elements,bending_pts,r,CON); % calculate conservative
forces and energies
F_D = f_diss_forces_mex(Links,vel,F_C); % calculate dissipative forces
F_TOT = F_C + F_D + F_APP; % new total forces [N]

U_LINKS(t,1) = U1; % links potential energy [J]
U_BENDING(t,1) = U2; % bending potential energy [J]
U_AREA_loc(t,1) = U3; % local area potential energy [J]
U_AREA_g(t,1) = ka*((AREA - A0t)^2)/(2*A0t); % global area potential energy

```



```

[J]
    U_VOLUME(t,1) = kv*((VOLUME - V0t)^2)/(2*V0t); % volume potential energy [J]
    U_TOT(t,1) = U_LINKS(t,1) + U_AREA_loc(t,1) + U_AREA_g(t,1) + U_VOLUME(t,1) +
    U_BENDING(t,1); % total potential energy [J]

    if U_TOT(t,1) > 1e-3 % error if potential energy is greater than threshold
    value

        n_error = t; %#ok<NASGU> % record timestep where error occurred
        f_save(save_flag,0); % save data to error dump
        error('Error: Invalid potential energy');

    end

    % Solve for New Velocities and Kinetic Energy:
    MASS = rho_hRBC*VOLUME/Nv; % update mass per node [kg]
    acc = (F_TOT_p + F_TOT)./MASS; % current accelerations [m/s^2]
    vel = vel_p + acc.*dt; % new velocities [m/s]
    F_TOT_p = F_TOT; % update previous forces [N]
    V_TOT(t,1) = f_ke_mex(vel,MASS); % total kinetic energy [J]

    % Update Connected Link Concentrations and Total Energy:
    VL = VOLUME*1e3; % update volume (in Liters) [L]
    CONC_SA_max = (num_avail/Na)/VL; % update current maximum concentration of
    links [mol/L], [M]
    CONC(57,1) = (sum(CON)/Na)/VL; % update current connected link concentration
    [mol/L], [M]
    CON_t(t,1) = sum(CON)/Ns; % record current connected link ratio (debugging)
    [unitless]
    E_TOT(t,1) = U_TOT(t,1) + V_TOT(t,1); % total energy [J]

    % Calculate Simulation Time and Output Progress:
    t_end = toc(tstart); % total elapsed time (real-time) at end of current
    timestep [s]
    n_error = nt; % record last step of simulation
    fprintf('After %d timesteps, the total potential energy is %e
    [J].\n',t,U_TOT(t,1))
    fprintf('The elapsed time is %d minute(s) and %f
    seconds.\n\n',floor(t_end/60),rem(t_end,60))

end

% Post-Simulation:
[ts_len,ts_ratio,t_sim_t,U_AVG,V_AVG,E_AVG,DT,DA,EI,r_deg,dL,pdeg6,Qa,Qh,RSA] =
f_pp(r,t_end,nt,dt,Elements,Links,LEN,U_TOT,V_TOT); % post processing calculations
n_end = t; % save final time step
[DATE_END] = f_save(save_flag,1); % save data

if plotting_flag == 1

f_plotting(nt,dt,U_LINKS,U_AREA_loc,U_AREA_g,U_VOLUME,U_BENDING,V_TOT,CON_t,CON_p,

```

```
r,r0,Fx,Xmin,Xmax,Ak,Elements,Links,LEN,CON,print_flag); % plotting of data and  
exporting of plots  
  
end
```

## A.1.12 hRBC\_Triangulation.m

```
% hRBC_Triangulation
%
% Generates a triangulation for the Combined hRBC model using
% the distmeshsurface function
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   none
%
% LOAD FILES:
%   none
%
% INPUTS:
%   none
%
% OUTPUTS:
%   none
%
% GLOBAL VARIABLES:
%   none
%
% SAVE FILES:
%   hRBC_Triangulation.mat

clear all
close all
clc

% Create Triangulation:
D0 = 7.82e-6; % hRBC diameter [m]
a0 = 0.00518; % shape equation constant 1 [unitless]
a1 = 2.0026; % shape equation constant 2 [unitless]
a2 = -4.491; % shape equation constant 3 [unitless]

bbox = [-4e-6,-4e-6,-4e-6;4e-6,4e-6,4e-6]; % extreme corners of bounding box (for
triangulation) [m]
h0 = 0.08108966e-6; % initial grid spacing in bounding box [m]

fd = @(p) D0^2*(1 - 4*(p(:,1).^2 + p(:,2).^2)/(D0^2)).*(a0 + a1*(p(:,1).^2 +
p(:,2).^2)/(D0^2) + a2*((p(:,1).^2 + p(:,2).^2)/(D0^4)).^2 - p(:,3).^2;
[p,tri] = distmeshsurface(fd,@huniform,h0,bbox); % perform triangulation

% Extract Data from Triangulation:
TR = TriRep(tri, p(:,1), p(:,2), p(:,3)); % create TriRep class
Links = edges(TR); % generate Links array
```

```

nodes = p; % assign p to nodes
Elements = tri; % assign tri to Elements
Nv = length(nodes); % number of nodes [unitless]
Nt = length(Elements); % number of elements [unitless]
Ns = length(Links); % number of links [unitless]

[TRILinks,bending_pts] = f_connectivity(Links,Elements); % generate TRILinks and
bending_pts matrices

% Evaluate Triangulation Quality:
% Vertex Degree Quality:
bc = histc(Links,unique(Links)); % count number of times each node appears in
Links array
deg = bc(:,1) + bc(:,2); % total degree of each node
dn = histc(deg,unique(deg)); % number of nodes of each degree
r_deg = [unique(deg) dn]; % table of vertex degree results

min_deg = min(deg); %#ok<NASGU> % minimum degree present
max_deg = max(deg); %#ok<NASGU> % maximum degree present
avg_deg = mean(deg); %#ok<NASGU> % average degree of all nodes
std_deg = std(deg); %#ok<NASGU> % standard deviation of all node degrees
avg_len = mean(LEN); % average length of all edges
std_len = std(LEN); % standard deviation of all edge lengths
dL = std_len/avg_len; % distribution of link length
pdeg6 = dn(unique(deg) == 6)/sum(dn); % percentage of degree 6 nodes

% Element Area Quality:
Q = zeros(Nt,1); % initialize element quality vector

for i = 1:1:Nt

    a = Elements(i,1);
    b = Elements(i,2);
    c = Elements(i,3);

    xa = r(a,1); % x-coordinate of node a
    ya = r(a,2); % y-coordinate of node a
    za = r(a,3); % z-coordinate of node a
    xb = r(b,1); % x-coordinate of node b
    yb = r(b,2); % y-coordinate of node b
    zb = r(b,3); % z-coordinate of node b
    xc = r(c,1); % x-coordinate of node c
    yc = r(c,2); % y-coordinate of node c
    zc = r(c,3); % z-coordinate of node c

    rab = [(xa - xb) (ya - yb) (za - zb)];
    rbc = [(xb - xc) (yb - yc) (zb - zc)];
    rca = [(xc - xa) (yc - ya) (zc - za)];

    Lab = sqrt(rab(1,1)*rab(1,1) + rab(1,2)*rab(1,2) + rab(1,3)*rab(1,3));
    Lbc = sqrt(rbc(1,1)*rbc(1,1) + rbc(1,2)*rbc(1,2) + rbc(1,3)*rbc(1,3));

```

```

    Lca = sqrt(rca(1,1)*rca(1,1) + rca(1,2)*rca(1,2) + rca(1,3)*rca(1,3));

    Q(i,1) = (Lbc + Lca - Lab)*(Lca + Lab - Lbc)*(Lab + Lbc - Lca)/(Lab*Lbc*Lca);
% triangle quality
    % % NOTE: Q >= 0.5 indicates triangle is of acceptable quality
    % % max(Q) = 1 (for an equilateral triangle)
    % % min(Q) = 0 (for a degenerate triangle)

end

Qa = sum(Q)/Nt; % arithmetic mean quality
Qh = Nt/sum(1./Q); % harmonic mean quality

% Gyration Tensor Calculations:
Gyr = zeros(3,3);
Nv = length(r);
rx = r(:,1) - cx;
ry = r(:,2) - cy;
rz = r(:,3) - cz;
Gyr(1,1) = sum(rx.*rx)/Nv;
Gyr(1,2) = sum(rx.*ry)/Nv;
Gyr(1,3) = sum(rx.*rz)/Nv;
Gyr(2,1) = sum(ry.*rx)/Nv;
Gyr(2,2) = sum(ry.*ry)/Nv;
Gyr(2,3) = sum(ry.*rz)/Nv;
Gyr(3,1) = sum(rz.*rx)/Nv;
Gyr(3,2) = sum(rz.*ry)/Nv;
Gyr(3,3) = sum(rz.*rz)/Nv;
[eVect,eVals] = eig(Gyr); %#ok<ASGLU>

L1 = eVals(1,1);
L2 = eVals(2,2);
L3 = eVals(3,3);
Rg = sqrt(L1^2 + L2^2 + L3^2); %#ok<NASGU>
asphericity = L3^2 - 0.5*(L1^2 + L2^2); %#ok<NASGU>
acylindricity = abs(L1^2 - L2^2); %#ok<NASGU>
RSA = sqrt((3/2)*((L1^4 + L2^4 + L3^4)/(L1^2 + L2^2 + L3^2)^2) - 1/2); % relative
shape anisotropy

% Save Triangulation Data:
DATE_hRBC_Triangulation = datestr(now,'mm_dd_yyyy__HH_MM_SS'); % date and time of
simulation completion
save('hRBC_Triangulation.mat','DATE_hRBC_Triangulation','bending_pts','Elements','
Links','Ns','Nt','Nv','nodes')

```

### A.1.13 vcross.m

```
function o = vcross(s,t) %#codegen
%
% Accelerated version of the cross product
%
% Produced in MATLAB by Stephen Oursler as a part of "A Proposed
% Mechanical-Metabolic Model of the Human Red Blood Cell".
%
% CALLED BY:
%   f_cons_forces.m
%
% LOAD FILES:
%   none
%
% INPUTS:
%   s: vector 1
%   t: vector 2
%
% OUTPUTS:
%   o: cross product of vector 1 and vector 2
%
% GLOBAL VARIABLES:
%   none
%
% SAVE FILES:
%   none

    o = [(s(1,2)*t(1,3) - s(1,3)*t(1,2)) (s(1,3)*t(1,1) - s(1,1)*t(1,3))
(s(1,1)*t(1,2) - s(1,2)*t(1,1))];

end
```

## Bibliography

- [1] H. M. Ezzeldin, "Multi-scale Modeling of soft matter: Gas Vesicles and Red Blood Cells," *Ph.D. Thesis, University of Maryland*, 2012.
- [2] M. Clinic, "Biological valve replacement," [Online]. Available: <http://www.mayoclinic.org/tests-procedures/heart-valve-surgery/multimedia/img-20039044>.
- [3] M. Clinic, "Mechanical valve replacement," [Online]. Available: <http://www.mayoclinic.org/tests-procedures/heart-valve-surgery/multimedia/img-20039477>.
- [4] A. M. Forsyth, J. Wan, P. D. Owrutsky, M. Abkarian and H. A. Stone, "Multiscale approach to link red blood cell dynamics, shear viscosity, and ATP release," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, no. 27, pp. 10986-10991, 2011.
- [5] J. Wan, A. M. Forsyth and H. A. Stone, "Red blood cell dynamics: from cell deformation to ATP release," *Integrative Biology*, vol. 3, no. 10, pp. 972-981, 2011.
- [6] M. Tomita, "Whole-cell simulation: a grand challenge of the 21st century," *TRENDS in Biotechnology*, vol. 19, no. 6, pp. 205-210, 2001.
- [7] The MathWorks Inc., "MATLAB Version 2012a," The MathWorks Inc., Natick, MA, 2012.
- [8] C. T. Lim, E. H. Zhou and S. T. Quek, "Mechanical models for living cells - a review," *Journal of Biomechanics*, vol. 39, no. 2, pp. 195-216, 2006.
- [9] N. Mohandas and P. G. Gallagher, "Red cell membrane: past, present, and future," *Blood*, vol. 112, no. 10, pp. 3939-3948, 2008.
- [10] N. Mohandas and E. Evans, "Mechanical Properties of the Red Cell Membrane in Relation to Molecular Structure and Genetic Defects," *Annual Reviews of Biophysics and Biomolecular Structure*, vol. 23, pp. 787-818, 1994.
- [11] P. W. Kuchel, "Current status and challenges in connecting models of erythrocyte metabolism to experimental reality," *Progress in Biophysics & Molecular Biology*, vol. 85, no. 2-3, pp. 325-342, 2004.
- [12] P. J. Mulquiney and P. W. Kuchel, "Model of 2,3-bisphosphoglycerate metabolism in the human erythrocyte based on detailed enzyme kinetic equations: equations and parameter refinement," *Biochemical Journal*, vol. 342, pp. 581-596, 1999.

- [13] P. J. Mulquiney and P. W. Kuchel, "Model of 2,3-bisphosphoglycerate metabolism in the human erythrocyte based on detailed enzyme kinetic equations: computer simulation and Metabolic Control Analysis," *Biochemical Journal*, vol. 342, pp. 597-604, 1999.
- [14] P. J. Mulquiney, W. A. Bubbs and P. W. Kuchel, "Model of 2,3-bisphosphoglycerate metabolism in the human erythrocyte based on detailed enzyme kinetic equations: in vivo kinetic characterization of 2,3-bisphosphoglycerate synthase/phosphatase using C and P NMR," *Biochemical Journal*, vol. 342, pp. 567-580, 1999.
- [15] P. J. Mulquiney and P. W. Kuchel, *Modelling Metabolism with Mathematica*, Boca Raton: CRC Press, 2003.
- [16] S. Chien, "Red Cell Deformability and its Relevance to Blood Flow," *Annual Reviews of Physiology*, vol. 49, pp. 177-192, 1987.
- [17] M. Zeitz and P. Sens, "Reversibility of red blood cell deformation," *Physical Review E*, vol. 85, no. 5, p. 051904, 2012.
- [18] O. K. Baskurt and H. J. Meiselman, "Blood Rheology and Hemodynamics," *Seminars in Thrombosis and Hemostasis*, vol. 29, no. 5, pp. 435-450, 2003.
- [19] S. A. Pandit, H. L. Scott, G. H. W. Lim, M. Wortis and R. Mukhopadhyay, *Soft Matter Volume 4: Lipid Bilayers and Red Blood Cells*, vol. 4, G. Gompper and M. Schick, Eds., Weinheim: Wiley-VHC Verlag GmbH & Co., 2008.
- [20] D. Stokes, "Architecture of the cytoskeleton in red blood cells," 2013. [Online]. Available: <http://www.med.nyu.edu/skirball-lab/stokeslab/rbc.html>.
- [21] D. Hartmann, "A multiscale model for red blood cell mechanics," *Biomechanics and Modeling in Mechanobiology*, vol. 9, no. 1, pp. 1-17, 2010.
- [22] J. C. Hansen, R. Skalak, S. Chien and A. Hoger, "Influence of Network Topology on the Elasticity of the Red Blood Cell Membrane Skeleton," *Biophysical Journal*, vol. 72, no. 5, pp. 2369-2381, 1997.
- [23] J. C. Hansen, R. Skalak, S. Chien and A. Hoger, "An Elastic Network Model Based on the Structure of the Red Blood Cell Membrane Skeleton," *Biophysical Journal*, vol. 70, no. 1, pp. 146-166, 1996.
- [24] A. J. Baines, "The spectrin-ankyrin-4.1-adducin membrane skeleton: adapting eukaryotic cells to the demands of animal life," *Protoplasma*, vol. 244, no. 1-4, pp. 99-131, 2010.



- [25] B. T. Stokke, A. Mikkelsen and A. Elgsaeter, "Spectrin, Human Erythrocyte Shapes, and Mechanochemical Properties," *Biophysical Journal*, vol. 49, no. 1, pp. 319-327, 1986.
- [26] N. M. Burton and L. J. Bruce, "Modelling the structure of the red cell membrane," *Biochemistry and Cell Biology*, vol. 89, no. 2, pp. 200-215, 2011.
- [27] N. Mohandas and J. A. Chasis, "Red-Blood-Cell Deformability, Membrane Material Properties and Shape-Regulation by Transmembrane, Skeletal and Cytosolic Proteins and Lipids," *Seminars in Hematology*, vol. 30, no. 3, pp. 171-192, 1993.
- [28] N. Mohandas, J. A. Chasis and S. B. Shohet, "The Influence of Membrane Skeleton on Red-Cell Deformability, Membrane Material Properties, and Shape," *Seminars in Hematology*, vol. 20, no. 3, pp. 225-242, 1983.
- [29] K. E. Kasza, A. C. Rowat, J. L. T. E. Angelini, C. P. Brangwynne, G. H. Koenderink and D. A. Weitz, "The cell as a material," *Current Opinion in Cell Biology*, vol. 19, no. 1, pp. 101-107, 2007.
- [30] G. Bao and S. Suresh, "Cell and molecular mechanics of biological materials," *Nature Materials*, vol. 2, no. 11, pp. 715-725, 2003.
- [31] D. Leckband and J. Israelachvili, "Intermolecular forces in biology," *Quarterly Reviews of Biophysics*, vol. 34, no. 2, pp. 105-267, 2001.
- [32] M. Dao, C. T. Lim and S. Suresh, "Mechanics of the human red blood cell deformed by optical tweezers," *Journal of the Mechanics and Physics of Solids*, vol. 51, no. 11-12, pp. 2259-2280, 2003.
- [33] M. Dao, C. T. Lim and S. Suresh, "Erratum to "Mechanics of the human red blood cell deformed by optical tweezers"," *Journal of the Mechanics and Physics of solids*, vol. 53, pp. 493-494, 2005.
- [34] D. E. Discher and P. Carl, "New Insights into Red Cell Network Structure, Elasticity, and Spectrin Unfolding - A Current Review," *Cellular & Molecular Biology Letter*, vol. 6, no. 3, pp. 593-606, 2001.
- [35] B. Schmidt and F. Fraternali, "Universal formulae for the limiting elastic energy of membrane networks," *Journal of the Mechanics and Physics of Solids*, vol. 60, no. 1, pp. 172-180, 2012.
- [36] Z. Peng, X. Li, I. V. Pivkin, M. Dao, G. E. Karniadakis and S. Suresh, "Lipid bilayer and cytoskeletal interactions in a red blood cell," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 110, no. 33, pp. 13356-13361, 2013.
- [37] P. Dimitrakopoulos, "Analysis of the variation in the determination of the shear modulus of the erythrocyte membrane: Effects of the constitutive law

- and membrane modeling," *Physical Review E*, vol. 85, no. 4, p. 041917, 2012.
- [38] S. Henon, G. Lenormand, A. Richert and F. Gallet, "A New Determination of the Shear Modulus of the Human Erythrocyte Membrane Using Optical Tweezers," *Biophysical Journal*, vol. 76, no. 2, pp. 1145-1151, 1999.
  - [39] C. T. Lim, M. Dao, S. Suresh, C. H. Sow and K. T. Chew, "Large deformation of living cells using laser traps," *Acta Materialia*, vol. 52, no. 7, pp. 1837-1845, 2004.
  - [40] G. Lenormand, S. Henon, A. Richert, J. Simeon and F. Gallet, "Direct Measurement of the Area Expansion and Shear Moduli of the Human Red Blood Cell Membrane Skeleton," *Biophysical Journal*, vol. 81, no. 1, pp. 43-56, 2001.
  - [41] C. T. Lim, M. Dao, S. Suresh, C. H. Sow and K. T. Chew, "Corrigendum to "Large deformation of living cells using laser traps"," *Acta Materialia*, vol. 52, no. 7, pp. 4065-4066, 2004.
  - [42] Y. Z. Yoon, J. Kotar, G. Yoon and P. Cicuta, "The nonlinear mechanical response of the red blood cell," *Physical Biology*, vol. 5, no. 3, p. 036007, 2008.
  - [43] J. P. Mills, L. Qie, M. Dao, C. T. Lim and S. Suresh, "Nonlinear Elastic and Viscoelastic Deformation of the Human Red Blood Cell with Optical Tweezers," *Mechanics & chemistry of biosystems: MCB*, vol. 1, no. 3, pp. 169-180, 2004.
  - [44] D. A. Fedosov, B. Caswell and G. E. Karniadakis, "Systematic coarse-graining of spectrin-level red blood cell models," *Computer Methods in Applied Mechanics and Engineering*, vol. 199, no. 29-32, pp. 1937-1948, 2010.
  - [45] S. Suresh, "Mechanical response of human red blood cells in health and disease: Some structure-property-function relationships," *Journal of Materials Research*, vol. 21, no. 8, pp. 1871-1877, 2006.
  - [46] D. J. Higham, "Modeling and Simulating Chemical Reactions," *Society for Industrial and Applied Mathematics*, vol. 50, no. 2, pp. 347-368, 2008.
  - [47] G. A. Holzapfel, *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*, West Sussex: John Wiley & Sons, LTD, 2000.
  - [48] D. Boal, *Mechanics of the Cell*, 2nd ed., Cambridge, UK: Cambridge University Press, 2012.
  - [49] D. Frenkel and B. Smit, *Understanding Molecular Simulation*, San Diego: Academic Press, 2002.

- [50] S. I. Sandler, *An Introduction to Applied Statistical Thermodynamics*, USA: John Wiley & Sons, Inc., 2011.
- [51] R. Blossey, *Computational Biology: A Statistical Mechanics Perspective*, Boca Raton: Chapman & Hall/CRC, 2006.
- [52] P. J. Flory, *Principles of Polymer Chemistry*, Ithaca: Cornell University Press, 1953.
- [53] M. Doi and S. F. Edwards, *The Theory of Polymer Dynamics*, New York: Oxford University Press Inc., 1986.
- [54] P. Johnson, "Biocurious - Wormlike chains," 2006. [Online]. Available: <http://biocurious.com/2006/07/04/wormlike-chains>.
- [55] R. W. Ogden, G. Saccomandi and I. Sgura, "On worm-like chain models within the three-dimensional continuum mechanics framework," *Proceedings of the Royal Society A*, vol. 462, pp. 749-768, 2005.
- [56] J. F. Marko and E. D. Siggia, "Stretching DNA," *Macromolecules*, vol. 28, pp. 8759-8770, 1995.
- [57] M. Ostoja-Starzewski, "Lattice models in micromechanics," *Applied Mechanics Reviews*, vol. 55, no. 1, pp. 35-60, 2012.
- [58] W. Wintz, R. Everaers and U. Seifert, "Mesh Collapse in Two-Dimensional Elastic Networks under Compression," *Journal de Physique I*, vol. 7, no. 9, pp. 1097-1111, 1997.
- [59] S. Hyun and S. Torquato, "Effective elastic and transport properties of regular honeycombs for all densities," *Journal of Materials Research*, vol. 15, no. 9, pp. 1985-1993, 2009.
- [60] G. R. Plaza, "Energy distribution in disordered elastic networks," *Physical Review E*, vol. 82, no. 3, p. 031902, 2010.
- [61] X. Liu and N. Liang, "Effective elastic moduli of triangular lattice material with defects," *Journal of the Mechanics and Physics of Solids*, vol. 60, no. 10, pp. 1722-1739, 2012.
- [62] M. Sheinman, C. P. Broedersz and F. C. MacKintosh, "Nonlinear effective-medium theory of disordered spring networks," *Physical Review E*, vol. 85, no. 2, p. 021801, 2012.
- [63] X. Mao, O. Stenull and T. C. Lubensky, "Effective-medium theory of a filamentous triangular lattice," *Physical Review E*, vol. 87, no. 4, p. 042601, 2013.
- [64] O. Farago and Y. Kantor, "Elasticity of Gaussian and nearly Gaussian phantom networks," *Physical Review E*, vol. 62, no. 5, pp. 6094-6102, 2000.

- [65] O. Farago and Y. Kantor, "Entropic elasticity of phantom percolation networks," *Europhysics Letters*, vol. 52, no. 4, pp. 413-419, 2000.
- [66] Z. Zhou, P. Y. Lai and B. Joos, "Rigorous solution for the elasticity of diluted Gaussian spring networks," *Physical Review E*, vol. 62, no. 5, pp. 7490-7493, 2000.
- [67] F. Tanaka and S. F. Edwards, "Viscoelastic Properties of Physically Cross-Linked Networks. Transient Network Theory.," *Macromolecules*, vol. 25, no. 5, pp. 1516-1523, 1992.
- [68] S. Roy and H. J. Qi, "Micromechanical model for elasticity of the cell cytoskeleton," *Physical Review E*, vol. 77, no. 6, p. 061916, 2008.
- [69] T. Omori, T. Ishikawa, D. Barthes-Biesel, A. Salsac, J. Walter, Y. Imai and T. Yamaguchi, "Comparison between spring network models and continuum constitutive laws: Application to the large deformation of a capsule in shear flow," *Physical Review E*, vol. 83, no. 4, p. 041918, 2011.
- [70] M. Assidi, F. D. Reis and J. F. Ganghoffer, "Equivalent mechanical properties of biological membranes from lattice homogenization," *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 4, no. 8, pp. 1833-1845, 2011.
- [71] M. J. Saxton, "The membrane skeleton of erythrocytes. A percolation model.," *Biophysical Journal*, vol. 57, no. 6, pp. 1167-1177, 1999.
- [72] C. Dubus and J. B. Fournier, "A Gaussian model for the membrane of red blood cells with cytoskeletal defects," *Europhysics Letters*, vol. 75, no. 1, pp. 181-187, 2006.
- [73] S. Feng and P. N. Sen, "Percolation on Elastic Networks: New Exponent and Threshold," *Physical Review Letters*, vol. 52, no. 3, pp. 216-219, 1984.
- [74] S. Feng, M. F. Thorpe and E. Garboczi, "Effective-medium theory of percolation on central-force elastic networks," *Physical Review B*, vol. 31, no. 1, pp. 276-280, 1985.
- [75] W. Tang and M. F. Thorpe, "Mapping between random central-force networks and random resistor networks," *Physical Review B*, vol. 36, no. 7, pp. 3798-3802, 1987.
- [76] W. Tang and M. F. Thorpe, "Percolation of elastic networks under tension," *Physical Review B*, vol. 37, no. 10, pp. 5539-5551, 1988.
- [77] M. F. Sykes and J. W. Essam, "Exact Critical Percolation Probabilities for Site and Bond Problems in Two Dimensions," *Journal of Mathematical Physics*, vol. 5, no. 8, pp. 1117-1127, 1964.
- [78] R. Zhang and F. L. H. Brown, "Cytoskeleton mediated effective elastic properties of model red blood cell membranes," *The Journal of Chemical*

Physics, vol. 129, no. 6, p. 065101, 2008.

- [79] P. O. Persson, "Mesh Generation for Implicit Geometries," 2005. [Online]. Available: <http://persson.berkeley.edu/thesis/>.
- [80] J. Li, M. Dao, C. T. Lim and S. Suresh, "Supplementary Materials of "Spectrin-Level Modeling of the Cytoskeleton and Optical Tweezers Stretching of the Erythrocyte"," [Online]. Available: [http://www.mit.edu/~mingdao/papers/Biophys\\_J\\_2005\\_Li\\_et\\_al\\_sup.pdf](http://www.mit.edu/~mingdao/papers/Biophys_J_2005_Li_et_al_sup.pdf).
- [81] P. O. Persson and G. Strang, "A Simple Mesh Generator in MATLAB," SIAM Review, vol. 46, no. 2, pp. 329-345, 2004.
- [82] R. Skalak, A. Tozeren, R. P. Zarda and S. Chien, "Strain Energy Function of Red Blood Cell Membranes," Biophysical Journal, vol. 13, no. 3, pp. 245-264, 1973.
- [83] G. H. W. Lim, M. Wortis and R. Mukhopadhyay, "Stomatocyte-discocyte-echinocyte sequence of the human red blood cell: Evidence for the bilayer-couple hypothesis from membrane mechanics," Proceedings of the National Academy of Sciences of the United States of America, vol. 99, no. 26, pp. 16766-16769, 2002.
- [84] S. Svetina, D. Kuzman, R. E. Waugh, P. Ziherl and B. Zeks, "The cooperative role of membrane skeleton and bilayer in the mechanical behavior of red blood cells," Bioelectrochemistry, vol. 62, no. 2, pp. 107-113, 2004.
- [85] S. K. Boey, D. H. Boal and D. E. Discher, "Simulations of the Erythrocyte Cytoskeleton at Large Deformation. 1. Microscopic Models," Biophysical Journal, vol. 75, no. 3, pp. 1573-1583, 1998.
- [86] D. E. Discher, D. H. Boal and S. K. Boey, "Simulations of the Erythrocyte Cytoskeleton at Large Deformation. 2. Micropipette Aspiration," Biophysical Journal, vol. 73, no. 3, pp. 1584-1597, 1998.
- [87] H. Li and G. Lykotrafitis, "Two-Component Coarse-Grained Molecular-Dynamics Model for the Human Erythrocyte Membrane," Biophysical Journal, vol. 102, no. 1, pp. 75-84, 2012.
- [88] L. G. Jiang, H. A. Wu, X. Z. Zhou and X. X. Wang, "Coarse-Grained Molecular Dynamics Simulations of a Red Blood Cell," Chinese Physical Letters, vol. 27, no. 2, p. 28704, 2010.
- [89] J. Li, M. Dao, C. T. Lim and S. Suresh, "Spectrin-Level Modeling of the Cytoskeleton and Optical Tweezers Stretching of the Erythrocyte," Biophysical Journal, vol. 88, no. 5, pp. 3707-3719, 2005.
- [90] Z. Peng, R. J. Asaro and Q. Zhu, "Multiscale simulation of erythrocyte membranes," Physical Review E, vol. 81, no. 3, p. 031904, 2010.

- [91] Z. Peng, R. J. Asaro and Q. Zhu, "Multiscale modelling of erythrocytes in Stokes flow," *Journal of Fluid Mechanics*, vol. 686, pp. 299-337, 2011.
- [92] I. V. Pivkin and G. E. Karniadakis, "Accurate Coarse-Grained Modeling of Red Blood Cells," *Physical Review Letters*, vol. 101, no. 11, p. 118105, 2008.
- [93] D. A. Fedosov, "Multiscale Modeling of Blood Flow and Soft Matter," Ph.D. Thesis, Brown University, 2010.
- [94] D. A. Fedosov, B. Caswell and G. E. Karniadakis, "A Multiscale Red Blood Cell Model with Accurate Mechanics, Rheology, and Dynamics," *Biophysical Journal*, vol. 98, no. 10, pp. 2215-2225, 2010.
- [95] M. Dao, J. Li and S. Suresh, "Molecularly based analysis of deformation of spectrin network and human erythrocyte," *Materials Science and Engineering C*, vol. 26, no. 8, pp. 1232-1244, 2006.
- [96] S. Meille and E. J. Garboczi, "Linear elastic properties of 2D and 3D models of porous materials made from elongated objects," *Modelling and Simulation in Materials Science and Engineering*, vol. 9, no. 5, pp. 371-390, 2001.
- [97] R. D. Groot and P. B. Warren, "Dissipative particle dynamics: Bridging the gap between atomistic and mesoscopic simulation," *The Journal of Chemical Physics*, vol. 107, no. 11, pp. 4423-4435, 1997.
- [98] P. Espanol, "Fluid Particle Model," *Physical Review E*, vol. 57, no. 3, pp. 2930-2948, 1998.
- [99] S. Paramore, G. S. Ayton, D. T. Mirijanian and G. A. Voth, "Extending a Spectrin Repeat Unit 1: Linear Force-Extension Response," *Biophysical Journal*, vol. 90, no. 1, pp. 92-100, 2006.
- [100] S. Paramore, G. S. Ayton and G. A. Voth, "Extending a Spectrin Repeat Unit 2 Rupture Behavior," *Biophysical Journal*, vol. 90, no. 1, pp. 101-111, 2006.
- [101] J. Li, G. Lykotrafitis, M. Dao and S. Suresh, "Cytoskeletal dynamics of human erythrocyte," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 12, pp. 4937-4942, 2007.
- [102] W. Helfrich, "Elastic properties of lipid bilayers: theory and possible experiments," *Zeitschrift für Naturforschung. Teil C: Biochemie, Biophysik, Biologie, Virologie*, vol. 28, no. 11, pp. 693-703, 1973.
- [103] M. Deserno, "Fluid lipid membranes - a primer (Notes on membrane theory)," [Online]. Available: [https://www.cmu.edu/biolphys/deserno/Deserno\\_science.html](https://www.cmu.edu/biolphys/deserno/Deserno_science.html).

- [104] M. Deserno, "Notes on Differential Geometry," 2004. [Online]. Available: [https://www.cmu.edu/biolphys/deserno/Deserno\\_science.html](https://www.cmu.edu/biolphys/deserno/Deserno_science.html).
- [105] E. Baesu, S. Kalyanam and M. Mocanu, "Quantitative Evaluation of Mechanical Properties of Cell Membranes: An Exact Solution," *Journal of Mechanics of Materials and Structures*, vol. 2, no. 6, pp. 1193-1203, 2007.
- [106] J. Lemaitre, *Handbook of Materials Behavior Models*, 1st ed., vol. 1, San Diego: Academic Press, 2001.
- [107] S. A. Safran, N. Gov, A. Nicolas, U. S. Schwarz and T. Tlusty, "Physics of cell elasticity, shape and adhesion," *Physica A*, vol. 352, no. 1, pp. 171-202, 2005.
- [108] L. Blanc, M. Salomao, X. Guo, X. An, W. Gratzer and N. Mohandas, "Control of Erythrocyte Membrane-Skeletal Cohesion by the Spectrin-Membrane Linkage," *Biochemistry*, vol. 49, no. 21, pp. 4516-4523, 2010.
- [109] R. I. Weed, P. L. LaCelle and E. W. Merrill, "Metabolic Dependence of Red Cell Deformability," *The Journal of Clinical Investigation*, vol. 48, no. 5, pp. 795-809, 1969.
- [110] P. Boivin, "Role of the phosphorylation of red blood cell membrane proteins," *Biochemical Journal*, vol. 256, no. 3, pp. 689-695, 1988.
- [111] S. Braunmuller, L. Schmid, E. Sackmann and T. Franke, "Hydrodynamic deformation reveals two coupled modes/time scales of red blood cell relaxation," *Soft Matter*, vol. 8, no. 44, pp. 11240-11248, 2012.
- [112] N. S. Gov and S. A. Safran, "Red Blood Cell Membrane Fluctuations and Shape Controlled by ATP-Induced Cytoskeletal Defects," *Biophysical Journal*, vol. 88, no. 3, pp. 1859-1874, 2005.
- [113] N. S. Gov, "Active elastic network: Cytoskeleton of the red blood cell," *Physical Review E*, vol. 75, no. 1, p. 011921, 2007.
- [114] L. Picas, F. Rico, M. Deforet and S. Scheuring, "Structural and Mechanical Heterogeneity of the Erythrocyte Membrane Reveals Hallmarks of Membrane Stability," *ACS Nano*, vol. 7, no. 2, pp. 1054-1063, 2013.
- [115] L. Picas, F. Rico, M. Deforet and S. Scheuring, "Supporting Information of Structural and Mechanical Heterogeneity of the Erythrocyte Membrane Reveals Hallmarks of Membrane Stability," 2013. [Online]. Available: [http://pubs.acs.org/doi/suppl/10.1021/nn303824j/suppl\\_file/nn303824j\\_si\\_001.pdf](http://pubs.acs.org/doi/suppl/10.1021/nn303824j/suppl_file/nn303824j_si_001.pdf).
- [116] C. Heussinger, "Stress relaxation through crosslink unbinding in cytoskeletal networks," *New Journal of Physics*, vol. 14, no. 9, p. 095029, 2012.

- [117] J. A. Chasis and N. Mohandas, "Erythrocyte Membrane Deformability and Stability: Two Distinct Membrane Properties That Are Independently Regulated by Skeletal Protein Associations," *Journal of Cell Biology*, vol. 103, no. 2, pp. 343-350, 1986.
- [118] T. Betz, M. Lenz, J. F. Joanny and C. Sykes, "ATP-dependent mechanics of red blood cells," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 36, pp. 15320-15325, 2009.
- [119] Y. Park, C. A. Best, T. Auth, N. S. Gov, S. A. Safran, G. Popescu, S. Suresh and M. S. Feld, "Metabolic remodeling of the human red blood cell membrane," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 4, pp. 1289-1294, 2010.
- [120] Y. Park, C. A. Best, T. Auth, N. S. Gov, S. A. Safran, G. Popescu, S. Suresh and M. S. Feld, "Metabolic remodeling of the human red blood cell membrane - Supporting Information," 2010. [Online]. Available: [http://www.pnas.org/content/suppl/2010/01/06/0910785107.DCSupplemental/pnas.0910785107\\_SI.pdf](http://www.pnas.org/content/suppl/2010/01/06/0910785107.DCSupplemental/pnas.0910785107_SI.pdf).
- [121] S. Manno, Y. Takakuwa, K. Nagao and N. Mohandas, "Modulation of Erythrocyte Membrane Mechanical Function by beta-Spectrin Phosphorylation and Dephosphorylation," *The Journal of Biological Chemistry*, vol. 270, no. 10, pp. 5659-5665, 1995.
- [122] J. Akola and R. O. Jones, "ATP Hydrolysis in Water - A Density Functional Study," *The Journal of Physical Chemistry B*, vol. 107, no. 42, pp. 11774-11783, 2003.
- [123] Y. Reichenberg and E. Altus, "Simulation of Erythrocyte's Deformation Using Conformational Changes in the Cytoskeleton," in *Proceedings of the 3rd International Conference on Cellular and Molecular Biology, Biophysics and Bioengineering*, Vouliagmeni, Athens, Greece, 2007.
- [124] G. Wolf, R. Bayer and D. Ostuni, "Stress-induced rigidification of erythrocytes as determined by laser diffraction and image analysis," *Optical Engineering*, vol. 31, no. 7, pp. 1475-1481, 1992.
- [125] T. M. Fischer, "Shape Memory of Human Red Blood Cells," *Biophysical Journal*, vol. 86, no. 5, pp. 3304-3313, 2004.
- [126] Q. Zhu and R. J. Asaro, "Spectrin Folding versus Unfolding Reactions and RBC Membrane Stiffness," *Biophysical Journal*, vol. 94, no. 7, pp. 2529-2545, 2008.
- [127] D. T. Mirjaniyan and G. A. Voth, "Unique elastic properties of the spectrin tetramer as revealed by multiscale coarse-grained modeling," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 4, pp. 1204-1208, 2008.



- [128] Q. Zhu, J. Zeng, M. S. Triantafyllou and D. K. P. Yue, "Direct Numerical Simulation of Single-Molecule DNA by Cable Dynamics," *Journal of Microelectromechanical Systems*, vol. 15, no. 5, pp. 1078-1087, 2006.
- [129] N. V. Kalyagina, M. V. Martinov and F. I. Ataullakhanov, "Mathematical Analysis of Human Red Blood Cell Volume Regulation with Regard to the Elastic Effect of the Erythrocyte Shell on Metabolic Processes," *Biochemistry (Moscow) Supplement Series: A Membrane and Cell Biology*, vol. 7, no. 2, pp. 122-133, 2013.
- [130] T. J. Larkin and P. W. Kuchel, "Mathematical Models of Naturally "Morphed" Human Erythrocytes: Stomatocytes and Echinocytes," *Bulletin of Mathematical Biology*, vol. 72, no. 6, pp. 1323-1333, 2010.
- [131] L. A. Sung and C. Vera, "Protofilament and Hexagon: A Three-Dimensional Mechanical Model for the Junction Complex in the Erythrocyte Membrane Skeleton," *Annals of Biomedical Engineering*, vol. 31, no. 11, pp. 1314-1326, 2003.
- [132] C. Vera, R. Skelton, F. Bossens and L. A. Sung, "3-D Nanomechanics of an Erythrocyte Junctional Complex in Equibiaxial and Anisotropic Deformations," *Annals of Biomedical Engineering*, vol. 33, no. 10, pp. 1387-1404, 2005.
- [133] Q. Zhu, C. Vera, R. J. Asaro, P. Sche and L. A. Sung, "A Hybrid Model for Erythrocyte Membrane: A Single Unit of Protein Network Coupled with Lipid Bilayer," *Biophysical Journal*, vol. 93, no. 2, pp. 386-400, 2007.
- [134] M. Oliveira, C. Vera, P. Valdez, Y. Sharma, R. Skelton and L. A. Sung, "Nanomechanics of Multiple Units in the Erythrocyte Membrane Skeletal Network," *Annals of Biomedical Engineering*, vol. 38, no. 9, pp. 2956-2967, 2010.
- [135] P. Sche, C. Vera and L. A. Sung, "Intertwined alpha-beta Spectrin Meeting Helical Actin Protofilament in the Erythrocyte Membrane Skeleton: Wrap-Around vs. Point-Attachement," *Annals of Biomedical Engineering*, vol. 39, no. 7, pp. 1984-1993, 2011.
- [136] M. P. Sheetz and S. J. Singer, "On the mechanism of ATP-induced shape changes in human erythrocyte membranes. 1. The role of the spectrin complex.," *The Journal of Cell Biology*, vol. 73, no. 3, pp. 638-646, 1977.
- [137] W. Birchmeier and S. J. Singer, "On the mechanism of ATP-induced shape changes in the human erythrocyte membranes. 2. The Role of ATP," *The Journal of Cell Biology*, vol. 73, no. 3, pp. 647-659, 1977.
- [138] D. E. Discher, R. Winardi, P. O. Schischmanoff, M. Parra and J. G. Conboy, "Mechanochemistry of Protein 4.1's Spectrin-Actin-Binding Domain: Ternary Complex Interactions, Membrane Binding, Network Integration, Structural Strengthening," *The Journal of Cell Biology*, vol.

- 130, no. 4, pp. 897-907, 1995.
- [139] V. Ohanian, L. C. Wolfe, K. M. John, J. C. Pinder and S. E. Lux, "Analysis of the Ternary Interaction of the Red Cell Membrane Skeletal Proteins Spectrin, Actin, and 4.1," *Biochemistry*, vol. 23, no. 19, pp. 4416-4420, 1984.
  - [140] E. Evans and K. Ritchie, "Dynamic Strength of Molecular Adhesion Bonds," *Biophysical Journal*, vol. 72, no. 4, pp. 1541-1555, 1997.
  - [141] C. Zhu, G. Bao and B. Wang, "Cell Mechanics: Mechanical Response, Cell Adhesion, and Molecular Deformation," *Annual Reviews of Biomedical Engineering*, vol. 2, no. 1, pp. 189-226, 2000.
  - [142] E. Evans, "Probing the Relation Between Force-Lifetime-and Chemistry in Single Molecular Bonds," *Annual Reviews of Biophysics and Biomolecular Structure*, vol. 30, no. 1, pp. 105-128, 2001.
  - [143] V. A. Lubarda and A. Marzani, "Viscoelastic response of thin membranes with application to red blood cells," *Acta Mechanica*, vol. 202, no. 1, pp. 1-16, 2009.
  - [144] V. A. Lubarda and A. Marzani, "Viscoelastic response of thin membranes with application to red blood cells: addendum," *Acta Mechanica*, vol. 212, no. 3-4, pp. 215-217, 2010.
  - [145] V. A. Lubarda, "Rate-Type Elasticity and Viscoelasticity of an Erythrocyte Membrane," *Journal of Mechanics of Materials and Structures*, vol. 6, no. 1-4, pp. 361-376, 2011.
  - [146] T. Kloppel and W. A. Wall, "A novel two-layer, coupled finite element approach for modeling the nonlinear elastic and viscoelastic behavior of human erythrocytes," *Biomechanics and Modeling in Mechanobiology*, vol. 10, no. 4, pp. 445-459, 2011.
  - [147] A. Bordbar, N. Jamshidi and B. O. Palsson, "iAB-RBC-283: A proteomically derived knowledge-base of erythrocyte metabolism that can be used to simulate its physiological and patho-physiological states," *BMC Systems Biology*, vol. 5, no. 110, pp. 1-12, 2011.
  - [148] H. X. Zhou, "Rate theories for biologists," *Quarterly Reviews of Biophysics*, vol. 43, no. 2, pp. 219-293, 2010.
  - [149] M. D. De Tullio, A. Cristallo, E. Balaras and R. Verzicco, "Direct numerical simulation of the pulsatile flow through an aortic bileaflet mechanical heart valve," *Journal of Fluid Mechanics*, vol. 622, pp. 259-290, 2009.
  - [150] T. Kruger, M. Gross, D. Raabe and F. Varnik, "Crossover from tumbling to tank-treading-like motion in dense simulated suspensions of red blood

- cells," *Soft Matter*, vol. 9, no. 37, pp. 9008-9015, 2013.
- [151] M. Nakamura, S. Bessho and S. Wada, "Spring-network-based model of a red blood cell for simulation mesoscopic blood flow," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 29, no. 1, pp. 114-128, 2013.
  - [152] A. V. Muravyov and I. A. Tikhomirova, "Role molecular signaling pathways in changes of red blood cell deformability," *Clinical Hemorheology and Microcirculation*, vol. 53, no. 1-2, pp. 45-59, 2013.
  - [153] P. H. Pawlowski, "Mechanosensitivity of cell membrane may govern creep-strain recovery, osmotic expansion and lysis," *Acta Biochimica Polonica*, vol. 56, no. 3, pp. 471-480, 2009.
  - [154] A. Bogdanova, A. Makhro, J. Wang, P. Lipp and L. Kaestner, "Calcium in Red Blood Cells - A Perilous Balance," *International Journal of Molecular Sciences*, vol. 14, no. 2, pp. 9848-9872, 2013.
  - [155] A. Dyrda, U. Cytlak, A. Ciuraszkiewicz, A. Lipinska, A. Cueff, G. Bouyer, S. Egee, P. Bennekou, V. L. Lew and S. L. Y. Thomas, "Local Membrane Deformations Activate Ca<sup>2+</sup>-Dependant K<sup>+</sup> and Anionic Currents in Intact Human Red Blood Cells," *PLoS ONE*, vol. 5, no. 2, p. e9447, 2010.
  - [156] M. Graupner, F. Erler and M. Meyer-Hermann, "A Theory of Plasma Membrane Calcium Pump Stimulation and Activity," *Journal of Biological Physics*, vol. 31, no. 2, pp. 183-206, 2005.
  - [157] C. Tanford, "Steady state of an ATP-driven calcium pump: Limitations on kinetic and thermodynamic parameters," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, no. 20, pp. 6161-6165, 1982.
  - [158] O. Woodford, "MATLAB Central File Exchange 22629-exportfig," 5 Dec 2013. [Online]. Available: <http://www.mathworks.com/matlabcentral/fileexchange/23629-exportfig>.